

ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC KINH TẾ ĐÀ NẴNG
KHOA THƯƠNG MẠI ĐIỆN TỬ

---o0o---



BÁO CÁO CUỐI KỲ MÔN: QUẢN TRỊ CƠ SỞ DỮ LIỆU

Đề tài:

**ĐỀ XUẤT XÂY DỰNG HỆ THỐNG GỢI Ý SẢN PHẨM
CHO ỨNG DỤNG THƯƠNG MẠI ĐIỆN TỬ
TIKI**

Thành viên

MSSV

Phan Xuân Hải Anh	:	221124029202
Vũ Phương Anh	:	221124029203
Dương Hoài Linh	:	221124029222
Nguyễn Văn Bảo Phúc	:	221124029232

Lớp: 48K29.2

GVHD: Hoàng Nguyên Vũ

Đà Nẵng, ngày 21 tháng 12 năm 2024

Mục lục

Giới thiệu đề tài	5
R1 + 2: Thu thập và đưa vào cơ sở dữ liệu	6
1. Tạo cơ sở dữ liệu để lưu trữ dữ liệu	6
2. Thu thập dữ liệu	9
R3: Tiền xử lý và tạo cơ sở dữ liệu mới	14
1. Tạo cơ sở dữ liệu mới	14
2. Tiền xử lý	18
2.1 Xóa các cột không cần thiết khỏi bảng Tiki_Product	18
2.2 Đổi định dạng ngày từ UnixTimestamp sang DateTime	19
2.3 Hàm kiểm tra sự tồn tại của giá trị NULL trong bình luận	19
2.4 Procedure: Xóa các dòng Null trong bình luận	19
2.5 Trigger: Kiểm tra sản phẩm trùng lặp	20
2.6 Trigger: Tự động xóa bình luận khi sản phẩm bị xóa	20
2.7 Trigger: Kiểm tra tính hợp lệ của bình luận	20
2.8 Trigger: Cập nhật tự động giá trị giảm giá khi giá sản phẩm thay đổi ...	21
R4: Sao lưu và khôi phục dữ liệu	22
1. Xác định loại và thời gian backup dữ liệu	22
2. Lưu trữ và quản lý backup	27
2.1 Xác định chiến lược backup	29
2.2 Kiểm tra tính toàn vẹn của backup	29
2.3 Giám sát backup qua Mail	30
2.4 Sao lưu backup tới thiết bị ngoài	32
2.5 Kiểm tra lịch sử backup	33
3. Phục hồi dữ liệu	34
R5: Phân quyền trên cơ sở dữ liệu	38
1. Ma trận phân quyền:	38
2. Phân quyền trên cơ sở dữ liệu:	38
R6: Trực quan hóa dữ liệu	41
1. Sử dụng Power BI	41
2. Sử dụng Tableau	41
3. Sử dụng Python	42
4. So sánh 3 công cụ sử dụng để trực quan hóa	43
R7: Xây dựng mô hình đề xuất sản phẩm	45
1. Giới Thiệu	45
2. Quá trình hoạt động của mô hình	45
2.1 Kết Nối Cơ Sở Dữ Liệu	45
2.2 Xử Lý Dữ Liệu	45
2.3 Vector Hóa Đặc Trưng	45
2.4 Thuật Toán KNN (K-Nearest Neighbors)	45
2.5 Tạo Giao Diện Tương Tác	46
3. Mã Python	46
4. Kết Luận:	50

Hình ảnh

Hình 1: Mô hình E-R đã thiết kế.....	7
Hình 2: Câu lệnh T-SQL để tạo cơ sở dữ liệu.....	7
Hình 3: Câu lệnh T-SQL để tạo các cột cho bảng Tiki_Product.....	7
Hình 4: Câu lệnh T-SQL để tạo các cột cho bảng Tiki_Customer.....	8
Hình 5: Câu lệnh T-SQL để tạo các cột cho bảng Tiki_Comment.....	8
Hình 6: Cơ sở dữ liệu ban đầu.....	8
Hình 7: Các thư viện dùng để thu thập dữ liệu.....	9
Hình 8: Câu lệnh để xác định các cột của cơ sở dữ liệu ban đầu.....	9
Hình 9: Câu lệnh để kết nối đến Microsoft SQL Server.....	10
Hình 10: Hàm để thêm dữ liệu vào cơ sở dữ liệu ban đầu.....	10
Hình 11: Hàm để thu thập dữ liệu sản phẩm từ API.....	11
Hình 12: Hàm để thu thập dữ liệu khách hàng từ API.....	11
Hình 13: Hàm để thu thập dữ liệu bình luận từ API.....	12
Hình 14: Câu lệnh tạo con trỏ.....	12
Hình 15: Danh sách các danh mục sản phẩm sẽ thu thập.....	12
Hình 16: Vòng lặp để thu thập dữ liệu.....	12
Hình 17: Câu lệnh thu thập dữ liệu về khách hàng và bình luận.....	13
Hình 19: Biến ‘cursor.commit()’.....	13
Hình 20: Sơ đồ E-R ban đầu.....	14
Hình 21: Sơ đồ Dữ liệu - Mọi quan hệ ban đầu.....	15
Hình 22: Tạo cơ sở dữ liệu mới.....	15
Hình 23: Tạo bảng Tiki_Product.....	16
Hình 24: Tạo bảng Tiki_Customer.....	16
Hình 25: Tạo bảng Tiki_Comment.....	16
Hình 26: Tạo bảng Tiki_Brand.....	17
Hình 27: Chèn dữ liệu vào bảng Tiki_Brand.....	17
Hình 28: Chèn dữ liệu vào bảng Tiki_Customer.....	17
Hình 29: Chèn dữ liệu vào bảng Tiki_Product.....	17
Hình 30: Chèn dữ liệu vào bảng Tiki_Comment.....	18
Hình 31: Thêm khóa ngoại cho bảng Tiki_Product.....	18
Hình 32: Sơ đồ Dữ liệu - Mọi quan hệ hoàn chỉnh.....	18
Hình 33: Xóa các cột không cần thiết.....	18
Hình 34: Đổi định dạng ngày.....	19
Hình 35: Hàm kiểm tra sự tồn tại của NULL.....	19
Hình 36: Xóa các dòng Null.....	19
Hình 37: Trigger kiểm tra sản phẩm trùng lặp.....	20
Hình 39: Trigger kiểm tra tính hợp lệ.....	21
Hình 41: Câu lệnh T-SQL tạo sao lưu.....	23
Hình 42: Câu lệnh T-SQL để lập lịch tự động tạo sao lưu.....	24
Hình 43: Cách sử dụng giao diện để tạo sao lưu.....	25
Hình 44: Cách sử dụng giao diện để tạo sao lưu.....	25
Hình 45: Cách sử dụng giao diện để lập lịch tự động.....	26
Hình 46: Các bản sao lưu đầu tiên.....	27
Hình 47: Nơi lưu trữ cục bộ.....	27

Hình 48: Nơi lưu trữ đám mây.	28
Hình 49: Thông tin các bản sao lưu ở đám mây.	28
Hình 50: Kiểm tra tính toàn vẹn của backup bằng giao diện	29
Hình 51: Kiểm tra tính toàn vẹn của backup bằng T-SQL	30
Hình 52: Cài đặt Mail.	30
Hình 53: Cài đặt Mail.	31
Hình 54: Cài đặt Mail.	31
Hình 55: Giám sát backup qua Mail.	32
Hình 56: Sao lưu backup tới thiết bị ngoài.	33
Hình 57: Kiểm tra lịch sử backup.	33
Hình 58: Lịch sử backup.	34
Hình 59: Các bước phục hồi dữ liệu.	35
Hình 60: Các bước phục hồi dữ liệu.	35
Hình 61: Các bước phục hồi dữ liệu.	36
Hình 62: Các bước phục hồi dữ liệu.	36
Hình 63: Các bước phục hồi dữ liệu.	37
Hình 15: Tạo Login cho người dùng.	39
Hình 64: Tạo Login cho người dùng.	39
Hình 65: Tạo User cho từng Login.	39
Hình 66: Tạo Role và phân quyền cho từng Role.	40
Hình 67: Thêm User vào Role.	40
Hình 68: Dashboard bằng Power BI.	41
Hình 69: Dashboard bằng Tableau.	41
Hình 70: Dashboard bằng Python.	42
Hình 71: Giao diện hệ thống đề xuất sản phẩm.	50

Giới thiệu đề tài

Trong bối cảnh thương mại điện tử ngày càng cạnh tranh, việc cung cấp trải nghiệm mua sắm cá nhân hóa cho người dùng đã trở thành một lợi thế cạnh tranh quan trọng. Hệ thống gợi ý sản phẩm, dựa trên các thuật toán học máy, đóng vai trò trung tâm trong việc nâng cao trải nghiệm người dùng và thúc đẩy doanh số.

Tiki, với tư cách là một trong những sàn thương mại điện tử hàng đầu tại Việt Nam, sở hữu một lượng lớn dữ liệu về sản phẩm và hành vi mua sắm của khách hàng. Việc khai thác hiệu quả nguồn dữ liệu này có thể hỗ trợ việc xây dựng các hệ thống gợi ý sản phẩm chính xác và hiệu quả cao.

Vì vậy, nhóm chúng em quyết định thực hiện đề tài “Xây dựng hệ thống gợi ý sản phẩm trên sàn thương mại điện tử Tiki”. Mục tiêu của đề tài này là phát triển một mô hình gợi ý sản phẩm dựa trên các thuật toán học máy tiên tiến, nhằm:

- Cá nhân hóa trải nghiệm người dùng: Đưa ra các gợi ý sản phẩm phù hợp với sở thích và nhu cầu của từng cá nhân.
- Tăng hiệu quả kinh doanh: Nâng cao tỷ lệ chuyển đổi, tăng giá trị đơn hàng trung bình và khuyến khích mua sắm lặp lại.
- Đóng góp vào nghiên cứu: Khám phá và áp dụng các kỹ thuật học máy mới vào lĩnh vực gợi ý sản phẩm trong bối cảnh thương mại điện tử Việt Nam.

R1 + 2: Thu thập và đưa vào cơ sở dữ liệu

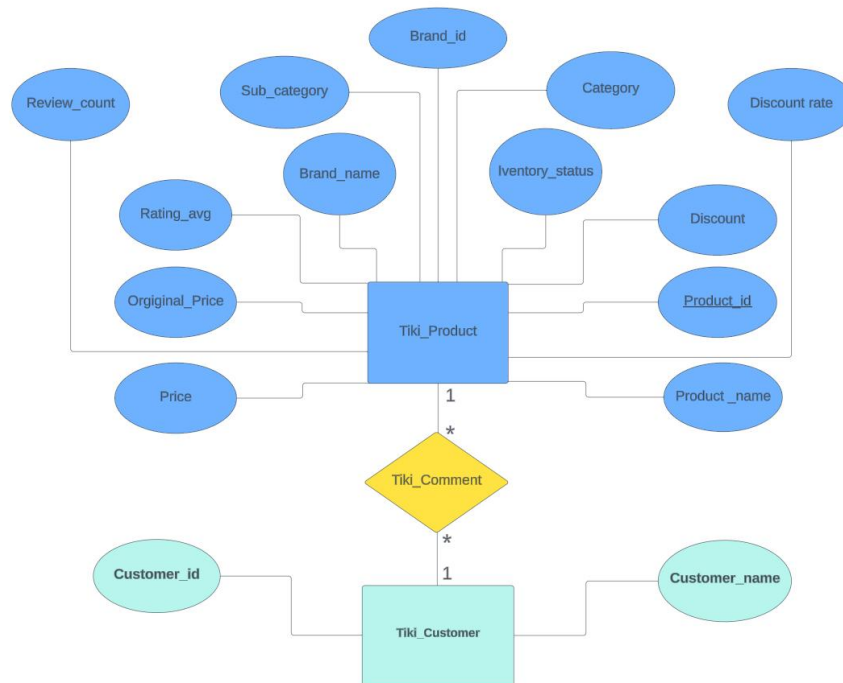
Nhóm thực hiện thu thập dữ liệu trên website chính thức của sàn thương mại điện tử Tiki (tiki.vn) theo phương pháp sử dụng thư viện requests trong Python để gửi yêu cầu HTTP tới API của Tiki nhằm lấy dữ liệu về sản phẩm và bình luận, sau đó lưu trữ vào cơ sở dữ liệu.

Quy trình thu thập dữ liệu như sau:

- Xác định các thông tin cần thiết có thể thu thập và tạo cơ sở dữ liệu ban đầu.
- Thu thập thông tin sản phẩm.
- Thu thập thông tin khách hàng.
- Thu thập bình luận của khách hàng về từng sản phẩm.
- Kết nối và đưa dữ liệu trực tiếp vào cơ sở dữ liệu.

1. Tạo cơ sở dữ liệu để lưu trữ dữ liệu

Sau khi xem xét chủ đề và các thông tin có thể thu thập từ trang chủ của Tiki như: ID sản phẩm, tên sản phẩm, giá, tên nhãn hàng, ID khách hàng, bình luận, đánh giá của khách hàng,... Nhóm tiến hành thiết kế mô hình thực thể - mối quan hệ (E-R Model) bằng công cụ Lucidchart. Việc khởi tạo các mô hình này giúp xác định rõ các thực thể có thể tạo ra từ dữ liệu, các thuộc tính của chúng cũng như mối quan hệ giữa các thực thể. Điều này đóng vai trò làm nền tảng cho quá trình triển khai cơ sở dữ liệu thực tế, đảm bảo hệ thống có cấu trúc rõ ràng, dễ quản lý và mở rộng trong tương lai. Ngoài ra, mô hình này còn hỗ trợ trực quan hóa hệ thống, giúp nhóm phát hiện và sửa chữa kịp thời các vấn đề tiềm ẩn trong thiết kế trước khi bắt đầu xây dựng cơ sở dữ liệu.



Hình 1: Mô hình E-R đã thiết kế.

Khi có được mô hình E-R, nhóm tiến hành xây dựng cơ sở dữ liệu trên phần mềm Microsoft SQL Sever để lưu những dữ liệu thu thập được theo các bước:

Đầu tiên, tạo cơ sở dữ liệu mới tên DATA:

```
CREATE DATABASE DATA
USE DATA
```

Hình 2: Câu lệnh T-SQL để tạo cơ sở dữ liệu.

Sau đó, tạo và kết nối các bảng:

```
CREATE TABLE Tiki_Product (
    Product_id INT PRIMARY KEY,
    Product_name NVARCHAR(300),
    Price MONEY,
    Original_price MONEY,
    Discount MONEY,
    Discount_rate FLOAT,
    Review_count INT,
    Rating_avg FLOAT,
    Order_count INT,
    Inventory_status VARCHAR(20),
    Brand_id VARCHAR(10),
    Brand_name NVARCHAR(100),
    Category NVARCHAR(100),
    Sub_category NVARCHAR(100)
)
```

Hình 3: Câu lệnh T-SQL để tạo các cột cho bảng Tiki_Product.

```
CREATE TABLE Tiki_Customer (
    Customer_id INT PRIMARY KEY,
    Customer_name NVARCHAR(100)
)
```

Hình 4: Câu lệnh T-SQL để tạo các cột cho bảng Tiki_Customer.

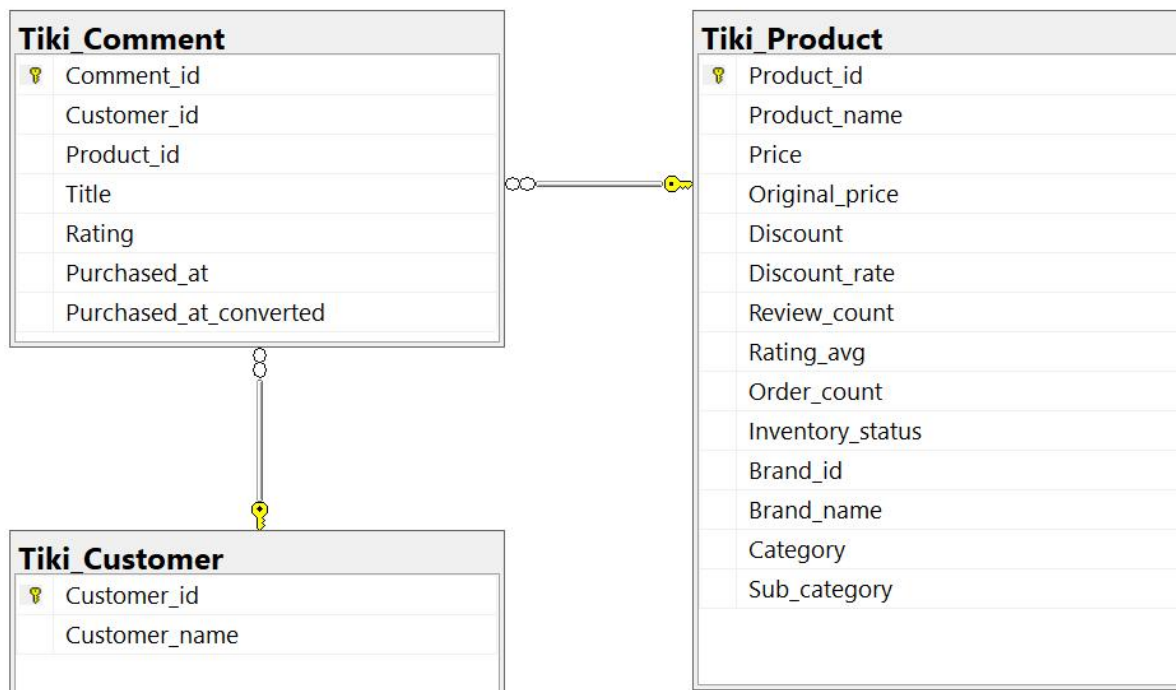
```
CREATE TABLE Tiki_Comment (
    Comment_id INT PRIMARY KEY IDENTITY(1,1),
    Customer_id INT,
    Product_id INT,
    Title NVARCHAR(300),
    Rating FLOAT,
    Purchased_at INT,
    CONSTRAINT fk_cus_id FOREIGN KEY(Customer_id) REFERENCES Tiki_Customer(Customer_id),
    CONSTRAINT fk_pid FOREIGN KEY(Product_id) REFERENCES Tiki_Product(Product_id)
)
```

Hình 5: Câu lệnh T-SQL để tạo các cột cho bảng Tiki_Comment.

Vì không có thông tin về Comment ID nên nhóm tạo khóa động để làm khóa chính cho bảng Comment.

=> **Kết quả thu được:**

Cơ sở dữ liệu ban đầu của nhóm gồm có 3 bảng: Tiki_Product (Lưu thông tin về sản phẩm), Tiki_Customer (Lưu thông tin về khách hàng) và Tiki_Comment (Lưu thông tin về bình luận sản phẩm của khách hàng).



Hình 6: Cơ sở dữ liệu ban đầu.

2. Thu thập dữ liệu

Sau khi có được cơ sở dữ liệu, nhóm tiến hành thu thập dữ liệu từ Tiki bằng cách dùng Python. Các bước được tiến hành lần lượt như bên dưới:

Đầu tiên, gọi các thư viện cần thiết:

```
import requests
import time
import random
```

Hình 7: Các thư viện dùng để thu thập dữ liệu.

Sau đó, viết các câu lệnh để thực hiện việc chèn dữ liệu vào cơ sở dữ liệu:

```
sql_insert_product = '''
    USE DATA
    INSERT INTO Tiki_Product
    VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
'''

sql_insert_customer = '''
    USE DATA
    INSERT INTO Tiki_Customer
    VALUES (?, ?)
'''

sql_insert_comment = '''
    USE DATA
    INSERT INTO Tiki_Comment
    VALUES (?, ?, ?, ?, ?)
'''
```

Hình 8: Câu lệnh để xác định các cột của cơ sở dữ liệu ban đầu.

Tiếp theo, nhóm viết hàm để thực hiện với việc kết nối với Microsoft SQL Server:

```

import pyodbc as odbc
DRIVER = 'SQL Server'
SERVER_NAME = 'DESKTOP-FR4PQFU\\SQL'
DATABASE_NAME = 'DATA'
USER_NAME = 'sa'
PASSWORD = 'Dh1@2001'
def connection_string(driver, server_name, database_name, username, password):
    conn_string = f"""
        DRIVER={{driver}};
        SERVER={server_name};
        DATABASE={database_name};
        username={username};
        password={password};
        Trust_Connection=yes;
    """
    return conn_string

try:
    conn = odbc.connect(connection_string(DRIVER, SERVER_NAME, DATABASE_NAME, USER_NAME, PASSWORD))
except odbc.DatabaseError as e:
    print('Database Error:')
    print(str(e.value[1]))
except odbc.Error as e:
    print('Connection Error:')
    print(str(e.value[1]))

```

Hình 9: Câu lệnh để kết nối đến Microsoft SQL Server.

Tại phần này, nhóm sử dụng thư viện ‘pyodbc’ để thiết lập kết nối với SQL Server. Sau đó, định nghĩa các tham số kết nối như: tên server, tên cơ sở dữ liệu và tài khoản đăng nhập. Tiếp theo, nhóm sử dụng ‘odbc.connect()’ để kết nối với cơ sở dữ liệu. Ngoài ra, nhóm có sử dụng cấu trúc ‘try-except’ để bắt lỗi nếu không thể kết nối với cơ sở dữ liệu. Bước tiếp theo, nhóm viết hàm 2 hàm ‘insert’, đây là hàm chính để thực hiện việc chèn dữ liệu vào cơ sở dữ liệu:

```

def insert_data(data, sql_insert_query, cursor):
    try:
        cursor.execute(sql_insert_query, data)
    except Exception as e:
        print(f"Lỗi insert dữ liệu: {e}")

def insert_data_customer(customer_data, insert_query, cursor):
    cursor.execute("SELECT * FROM Tiki_Customer WHERE Customer_id = %s" %(customer_data[0],))
    existing_customer = cursor.fetchone()
    if existing_customer:
        pass
    else:
        cursor.execute(insert_query, customer_data)

```

Hình 10: Hàm để thêm dữ liệu vào cơ sở dữ liệu ban đầu.

Ở đây, việc thêm dữ liệu vào bảng ‘Tiki_Customer’ sẽ được thực hiện bởi một hàm riêng vì nhóm nhận thấy có nhiều thông tin khách hàng bị trùng lặp khi thu thập dữ liệu, vì vậy để tránh việc xung đột với khóa chính thì nhóm đã viết hàm riêng để kiểm tra sự tồn tại của ‘Customer_id’, nếu thông tin ấy đã tồn tại trong bảng thì sẽ không chèn lại vào bảng để tránh việc trùng lặp.

Tiếp theo, nhóm lần lượt viết ba hàm ‘parser’ để thu thập dữ liệu từ API của Tiki, các hàm này sẽ làm nhiệm vụ chuyển JSON thu được từ API thành các dictionary:

```
def parser_product(json):
    d = dict()
    d['product_id'] = json.get('id')
    d['product_name'] = json.get('name')
    d['price'] = json.get('price')
    d['original_price'] = json.get('original_price')
    d['discount'] = json.get('discount')
    d['discount_rate'] = json.get('discount_rate')
    try:
        d['review_count'] = json.get('review_count')
    except AttributeError:
        d['review_count'] = 0
    try:
        d['rating_avg'] = json.get('rating_average')
    except AttributeError:
        d['rating_avg'] = 0
    try:
        d['order_count'] = json.get('quantity_sold').get('value')
    except AttributeError:
        d['order_count'] = 0
    d['inventory_status'] = json.get('inventory_status')
    try:
        d['brand_id'] = json.get('brand').get('id')
    except AttributeError:
        d['brand_id'] = None
    try:
        d['brand_name'] = json.get('brand').get('name')
    except AttributeError:
        d['brand_name'] = None
    d['category'] = json['breadcrumbs'][0]['name']
    d['sub_category'] = json['breadcrumbs'][1]['name']
    return d
```

Hình 11: Hàm để thu thập dữ liệu sản phẩm từ API.

```
def customer_parser(json):
    d = dict()
    d['customer_id'] = json.get('customer_id')
    try:
        d['customer_name'] = json.get('created_by').get('name')
    except AttributeError:
        d['customer_name'] = None
    return d
```

Hình 12: Hàm để thu thập dữ liệu khách hàng từ API.

```
def comment_parser(json):
    d = dict()
    d['customer_id'] = json.get('customer_id')
    d['product_id'] = json.get('product_id')
    d['title'] = json.get('title')
    d['rating'] = json.get('rating')
    try:
        d['purchased_at'] = json.get('created_by').get('purchased_at')
    except AttributeError:
        d['purchased_at'] = None
    return d
```

Hình 13: Hàm để thu thập dữ liệu bình luận từ API.

Trước khi, vào phần thu thập và chèn dữ liệu chính của đoạn code, nhóm sẽ tạo con trỏ (cursor) để tương tác với cơ sở dữ liệu:

```
cursor = conn.cursor()
```

Hình 14: Câu lệnh tạo con trỏ.

Để việc thu thập dữ liệu diễn ra nhanh hơn, nhóm thực hiện lấy các ID danh mục sản phẩm của Tiki, mỗi danh mục sẽ được dùng để gửi yêu cầu API lấy các sản phẩm thuộc danh mục đó:

```
list_category_id_1 = {1883, 1789, 2549, 1815, 1882, 1520}
list_category_id_2 = {8594, 931, 4384, 1975, 915, 1846, 1686}
list_category_id_3 = {4221, 1703, 1801, 27498, 8371, 6000, 15078}
```

Hình 15: Danh sách các danh mục sản phẩm sẽ thu thập.

Cuối cùng, nhóm thực hiện vòng lặp để thu thập dữ liệu. Trước hết là vòng lặp để thu thập thông tin sản phẩm:

```
for c_id in list_category_id_1:
    for i in range(1, 41):
        params = {'limit': '40', 'page': str(i), 'category': str(c_id)}
        print('page',[i], 'category',[c_id])
        response = requests.get('https://tiki.vn/api/personalish/v1/blocks/listings', headers=headers, params=params)
        if response.status_code == 200:
            for record in response.json().get('data'):
                pid = record.get('id')
                response1 = requests.get(f'https://tiki.vn/api/v2/products/{pid}', headers=headers, timeout=10)
                if response1.status_code == 200:
                    product_data = parser_product(response1.json())
                    product_data = list(product_data.values())
                    insert_data(product_data, sql_insert_product, cursor)
```

Hình 16: Vòng lặp để thu thập dữ liệu.

Vòng lặp này sẽ lặp qua từng ID danh mục sản phẩm trong các danh sách danh mục sản phẩm đã tạo. Tại mỗi danh mục sẽ lặp qua 40 trang dữ liệu, mỗi trang chứa 40 sản phẩm. Sau đó gửi yêu cầu ‘GET’ đến API ‘https://tiki.vn/api/personalish/v1/blocks/listings’ để lấy thông tin về ID sản phẩm. Nếu API trả về thành công (status_code = 200), lấy danh sách sản phẩm trong JSON. Để lấy thông tin về từng sản phẩm cụ thể, lấy ID của từng sản phẩm trong danh sách vừa lấy và gọi API thứ hai ‘https://tiki.vn/api/v2/products/{pid}’. Tiếp đó, tiến hành lưu thông tin sản phẩm vào cơ sở dữ liệu.

Tương tự như vậy, trong từng trang của 1 sản phẩm, tiến hành thu thập dữ liệu về khách hàng và bình luận về sản phẩm, ở đây nhóm sẽ thu thập thông tin về khách hàng và bình luận trên 10 trang bình luận, mỗi trang chứa 5 bình luận:

```
print('Crawl comment for product {}'.format(pid))
for j in range(1,11):
    params2 = {'limit': '5', 'page': str(j), 'spid': pid, 'product_id': pid}
    response = requests.get('https://tiki.vn/api/v2/reviews', headers=headers, params=params2, timeout=10)
    if response.status_code == 200:
        for comment in response.json().get('data'):
            customer_data = customer_parser(comment)
            customer_data = list(customer_data.values())

            comment_data = comment_parser(comment)
            comment_data = list(comment_data.values())

            insert_data_customer(customer_data, sql_insert_customer, cursor)
            insert_data(comment_data, sql_insert_comment, cursor)
```

Hình 17: Câu lệnh thu thập dữ liệu về khách hàng và bình luận.

Trong vòng lặp về thu thập dữ liệu, nhóm có sử dụng ‘timesleep’ sau mỗi lần hoàn thành lấy bình luận của 1 sản phẩm:

```
time.sleep(random.randrange(0, 2))
```

Hình 18: Dừng giữa các lần thu thập.

Ở đây, sau mỗi vòng lặp, chương trình sẽ dừng ngẫu nhiên trong khoảng 0 đến 2 giây, điều này để tránh việc bị chặn API.

Ngoài ra để tránh việc mất mát dữ liệu khi có sự cố, nhóm đặt 2 biến ‘cursor.commit()’:

```
cursor.commit()
```

Hình 19: Biến ‘cursor.commit()’.

Biến thứ nhất được đặt sau khi hoàn thành việc chèn dữ liệu cho 1 sản phẩm và bình luận về sản phẩm ấy, điều này lưu các thay đổi vào cơ sở dữ liệu, nếu có sự cố xảy ra trong các sản phẩm tiếp theo, dữ liệu của các sản phẩm trước đã được lưu an toàn.

Biến thứ hai được để ở ngoài cùng để đảm bảo dữ liệu đã được lưu hoàn chỉnh vào cơ sở dữ liệu.

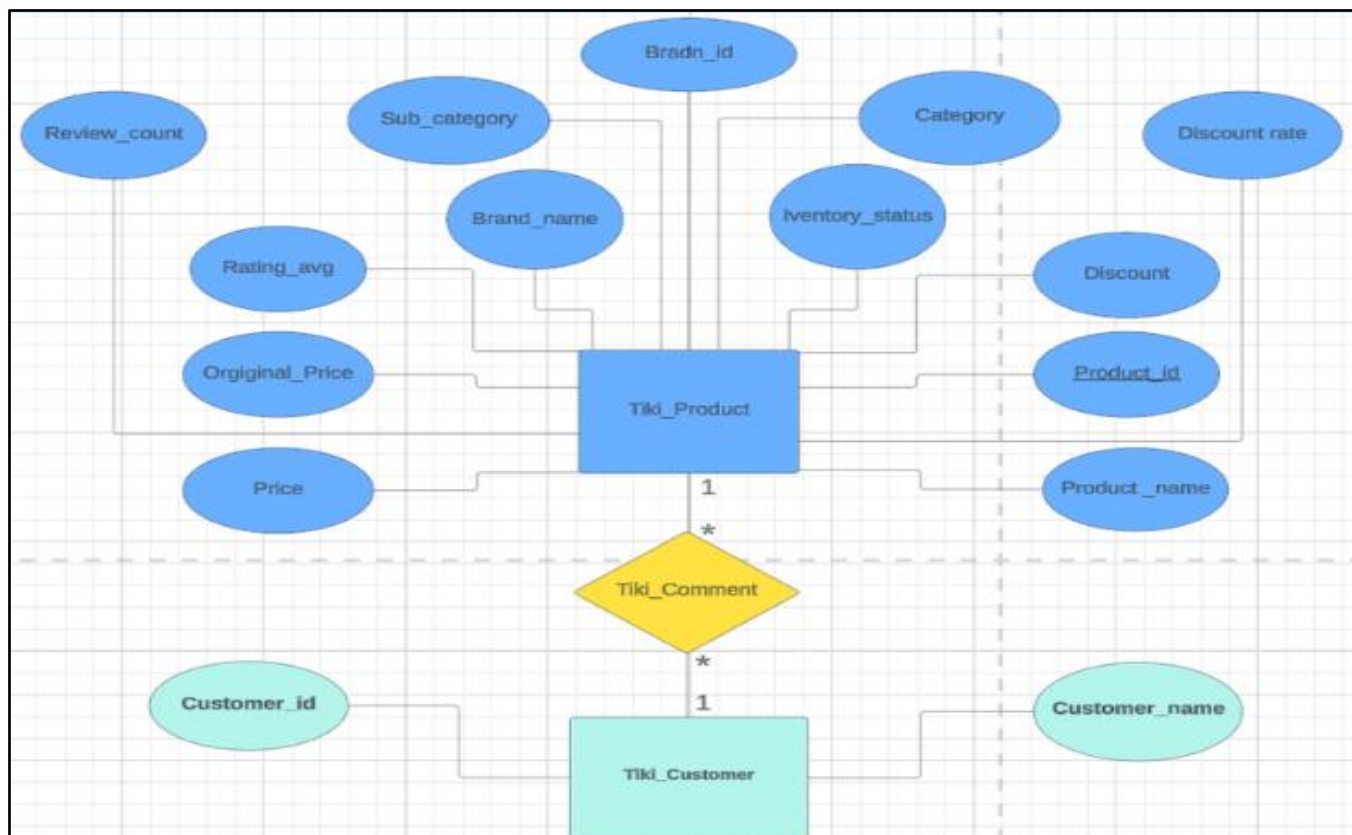
R3: Tiền xử lý và tạo cơ sở dữ liệu mới

1. Tạo cơ sở dữ liệu mới

Ban đầu, nhóm đã thu thập dữ liệu về bảng SQL server với cấu trúc gồm: 3 bảng: **Tiki_Product**, **Tiki_Comment**, **Tiki_Customer**.

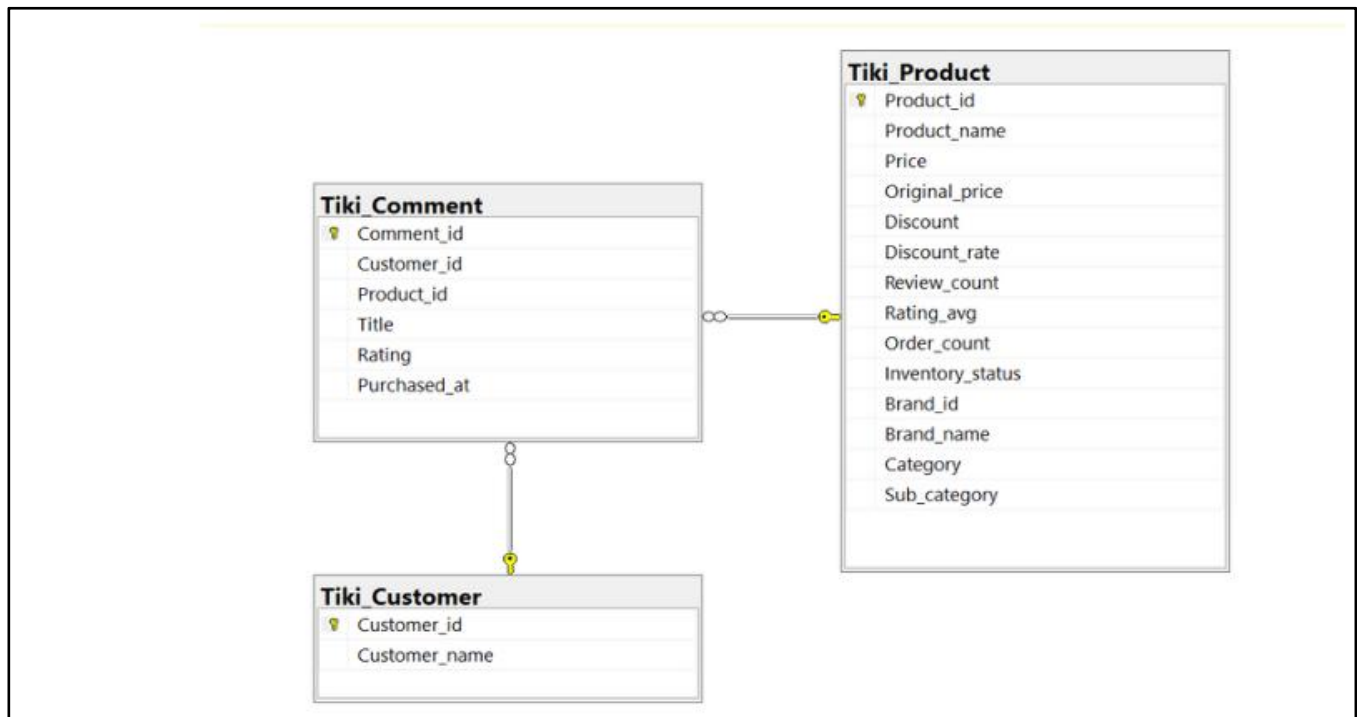
Mối quan hệ:

- **Tiki_Product** có quan hệ 1-N với **Tiki_Comment** (một sản phẩm có nhiều bình luận).
- **Tiki_Customer** có quan hệ 1-N với **Tiki_Comment** (một khách hàng có thể viết nhiều bình luận).
- **Tiki_Comment** là cầu nối giữa **Tiki_Product** và **Tiki_Customer** trong mối quan hệ nhiều - nhiều.



Hình 20: Sơ đồ E-R ban đầu.

Sau khi có Sơ đồ Thực thể - Mối quan hệ thì nhóm đã tạo được Sơ đồ Dữ liệu - Mối quan hệ như sau:



Hình 21: Sơ đồ Dữ liệu - Mối quan hệ ban đầu.

Đầu tiên, nhóm sẽ tạo một cơ sở dữ liệu mới có tên DATA_FOR_ML để lưu trữ thông tin dữ liệu đã thu thập được.

```
--Tạo CSDL mới
CREATE DATABASE DATA_FOR_ML;
USE DATA_For_ML;
```

Hình 22: Tạo cơ sở dữ liệu mới.

Tiếp theo, nhóm sẽ đi tạo các bảng chứa dữ liệu cần thiết từ dữ liệu thu thập được.

- Bảng **Tiki_Product**:
 - + Bảng này sẽ lưu thông tin về sản phẩm, bao gồm ID, tên, giá cả, khuyến mãi, đánh giá trung bình, tồn kho, thương hiệu, danh mục và phân loại phụ.
 - + Khóa chính: Product_id.

```

CREATE TABLE Tiki_Product (
    Product_id INT PRIMARY KEY,
    Product_name NVARCHAR(300),
    Price MONEY,
    Original_price MONEY,
    Discount MONEY,
    Discount_rate FLOAT,
    Rating_avg FLOAT,
    Inventory_status VARCHAR(20),
    Brand_id VARCHAR(10),
    Brand_name NVARCHAR(100),
    Category NVARCHAR(100),
    Sub_category NVARCHAR(100)
)

```

Hình 23: Tạo bảng Tiki_Product.

- Bảng **Tiki_Customer**:
 - + Bảng này sẽ lưu thông tin khách hàng như ID và tên.
 - + Khóa chính: Customer_id.

```

CREATE TABLE Tiki_Customer (
    Customer_id INT PRIMARY KEY,
    Customer_name NVARCHAR(100)
)

```

Hình 24: Tạo bảng Tiki_Customer.

- Bảng **Tiki_Comment**:
 - + Bảng này lưu thông tin các bình luận, đánh giá từ khách hàng về sản phẩm.
 - + Khóa chính: Comment_id (tự động tăng).
 - + Khóa ngoại:
 - Customer_id tham chiếu đến bảng Tiki_Customer.
 - Product_id tham chiếu đến bảng Tiki_Product.

```

CREATE TABLE Tiki_Comment (
    Comment_id INT PRIMARY KEY IDENTITY(1,1),
    Customer_id INT,
    Product_id INT,
    Title NVARCHAR(300),
    Rating FLOAT,
    Purchased_at_old INT,
    CONSTRAINT fk_cus_id FOREIGN KEY(Customer_id) REFERENCES Tiki_Customer(Customer_id),
    CONSTRAINT fk_pid FOREIGN KEY(Product_id) REFERENCES Tiki_Product(Product_id)
)

```

Hình 25: Tạo bảng Tiki_Comment

- Bảng **Tiki_Brand**:
 - + Bảng này lưu thông tin thương hiệu (nhãn hiệu) của các sản phẩm.
 - + Khóa chính: Brand_id.

```
CREATE TABLE Tiki_Brand (
    Brand_id VARCHAR(10) PRIMARY KEY,
    Brand_name NVARCHAR(100)
);
```

Hình 26: Tạo bảng Tiki_Brand

Sau đó, nhóm sẽ chèn dữ liệu vào các bảng vừa tạo phía trên.

- Bảng **Tiki_Brand**: Chèn dữ liệu về nhãn hiệu từ bảng Tiki_Product trong CSDL cũ vào bảng Tiki_Brand, loại bỏ các giá trị trùng lặp bằng DISTINCT.

```
INSERT INTO Tiki_Brand(Brand_id,Brand_name)
SELECT DISTINCT Brand_id, Brand_name
FROM [DATA].[dbo].[Tiki_Product];
```

Hình 27: Chèn dữ liệu vào bảng Tiki_Brand

- Bảng **Tiki_Customer**: Chèn dữ liệu khách hàng từ bảng Tiki_Customer trong CSDL cũ vào bảng mới.

```
-- Chèn dữ liệu vào bảng Tiki_Customer
INSERT INTO Tiki_Customer (Customer_id, Customer_name)
SELECT Customer_id, Customer_name
FROM [DATA].[dbo].[Tiki_Customer];
```

Hình 28: Chèn dữ liệu vào bảng Tiki_Customer

- Bảng **Tiki_Product**: Chèn dữ liệu sản phẩm từ bảng Tiki_Product trong CSDL cũ vào bảng mới.

```
INSERT INTO Tiki_Product (Product_id, Product_name, Price, Original_price, Discount, Discount_rate,
    Rating_avg, Inventory_status, Brand_id, Category, Sub_category)
SELECT Product_id, Product_name, Price, Original_price, Discount, Discount_rate,
    Rating_avg, Inventory_status, Brand_id, Category, Sub_category
FROM [DATA].[dbo].[Tiki_Product];
```

Hình 29: Chèn dữ liệu vào bảng Tiki_Product

- Bảng **Tiki_Comment**:
 - + Bật IDENTITY_INSERT để cho phép chèn giá trị vào cột tự động tăng Comment_id.
 - + Chèn dữ liệu từ bảng Tiki_Comment trong CSDL cũ.
 - + Tắt IDENTITY_INSERT sau khi hoàn thành.

```

SET IDENTITY_INSERT Tiki_Comment ON;
INSERT INTO Tiki_Comment (Comment_id, Customer_id, Product_id, Title, Rating, Purchased_at_old)
SELECT Comment_id, Customer_id, Product_id, Title, Rating, Purchased_at
FROM [DATA].[dbo].[Tiki_Comment];
SET IDENTITY_INSERT Tiki_Comment OFF;

```

Hình 30: Chèn dữ liệu vào bảng Tiki_Comment

Cuối cùng, nhóm thêm khóa ngoại cho bảng **Tiki_Product**, thiết lập khóa ngoại Brand_id trong bảng Tiki_Product để tham chiếu đến bảng Tiki_Brand. Điều này sẽ đảm bảo tính liên kết dữ liệu giữa sản phẩm và nhãn hiệu.

```

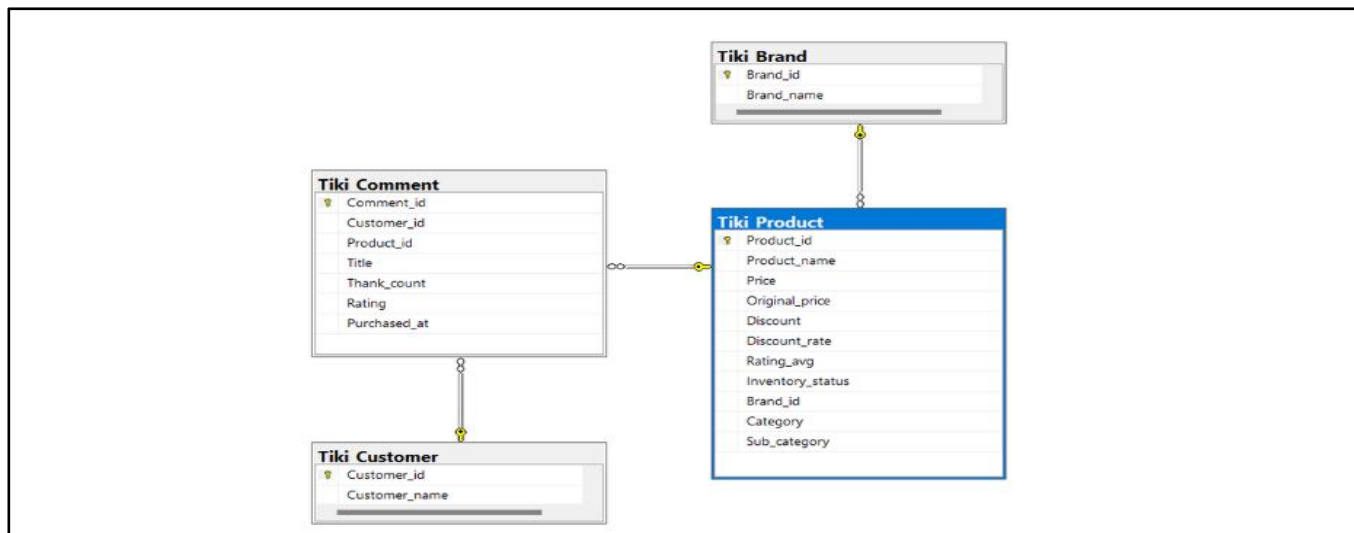
ALTER TABLE Tiki_Product
ADD CONSTRAINT FK_Tiki_Product_Brand
FOREIGN KEY (Brand_id) REFERENCES Tiki_Brand(Brand_id)

```

Hình 31: Thêm khóa ngoại cho bảng Tiki_Product.

=> Kết quả:

Nhóm có CSDL hoàn chỉnh:



Hình 32: Sơ đồ Dữ liệu - Mối quan hệ hoàn chỉnh

2. Tiền xử lý

2.1 Xóa các cột không cần thiết khỏi bảng Tiki_Product

```

ALTER TABLE Tiki_Product
DROP COLUMN Brand_name;

```

Hình 33: Xóa các cột không cần thiết

2.2 Đổi định dạng ngày từ UnixTimestamp sang DateTime

```
--Đổi ngày từ UnixTimestamp sang DateTime
ALTER TABLE [DATA_For_ML].[dbo].[Tiki_Comment]
ADD Purchased_at DATETIME;

UPDATE [DATA_For_ML].[dbo].[Tiki_Comment]
SET Purchased_at = FORMAT(DATEADD(SECOND, CAST(Purchased_at_old AS BIGINT), '1970-01-01'), 'yyyy-MM-dd HH:mm:ss');

ALTER TABLE [DATA_For_ML].[dbo].[Tiki_Comment]
DROP COLUMN Purchased_at_old;
```

Hình 34: Đổi định dạng ngày

2.3 Hàm kiểm tra sự tồn tại của giá trị NULL trong bình luận

```
--Kiểm tra giá trị null
CREATE FUNCTION CheckForNullValue()
RETURNS BIT
AS
BEGIN
    DECLARE @has_null BIT;

    SELECT
        @has_null = CASE WHEN COUNT(*) > 0 THEN 1 ELSE 0 END
    FROM
        Tiki_Comment
    WHERE
        Title IS NULL
        OR Rating IS NULL
        OR Purchased_at IS NULL;

    RETURN @has_null;
END;
```

Hình 35: Hàm kiểm tra sự tồn tại của NULL

2.4 Procedure: Xóa các dòng Null trong bình luận

```
---Xóa các giá trị null
CREATE PROCEDURE DeleteRowsWithNull
AS
BEGIN
    DELETE FROM Tiki_Comment
    WHERE Title IS NULL
        OR Rating IS NULL
        OR Purchased_at IS NULL;
END;
```

Hình 36: Xóa các dòng Null

2.5 Trigger: Kiểm tra sản phẩm trùng lặp

```
--Kiểm tra sản phẩm trùng lặp
CREATE TRIGGER CheckUniqueProductName
ON Tiki_Product
AFTER INSERT
AS
BEGIN
    IF EXISTS (
        SELECT 1
        FROM inserted
        WHERE Product_name = inserted.Product_name
            AND Brand_name = inserted.Brand_name
    )
    BEGIN
        RAISERROR('Error: Product name must be unique within the same brand.', 16, 1)
        ROLLBACK TRANSACTION
    END
END
```

Hình 37: Trigger kiểm tra sản phẩm trùng lặp

2.6 Trigger: Tự động xóa bình luận khi sản phẩm bị xóa

```
--Xóa bình luận sau khi xóa sản phẩm
CREATE TRIGGER DeleteCommentsAfterProductDelete
ON Tiki_Product
AFTER DELETE
AS
BEGIN
    DELETE FROM Tiki_Comment
    WHERE Product_id IN (SELECT Product_id FROM deleted)
END
```

Hình 38: Trigger tự động xóa bình luận khi sản phẩm bị xóa

2.7 Trigger: Kiểm tra tính hợp lệ của bình luận

```

--Kiểm tra bình luận hợp lệ
CREATE TRIGGER ValidateCommentData
ON Tiki_Comment
AFTER INSERT
AS
BEGIN
    IF EXISTS (
        SELECT 1
        FROM inserted
        WHERE Rating < 1 OR Rating > 5
    )
    BEGIN
        RAISERROR('Error: Rating must be between 1 and 5.', 16, 1)
        ROLLBACK TRANSACTION
    END

    IF EXISTS (
        SELECT 1
        FROM inserted
        WHERE NOT EXISTS (
            SELECT 1
            FROM Tiki_Customer
            WHERE Customer_id = inserted.Customer_id
        )
    )
    BEGIN
        RAISERROR('Error: Customer does not exist.', 16, 1)
        ROLLBACK TRANSACTION
    END

    IF EXISTS (
        SELECT 1
        FROM inserted
        WHERE NOT EXISTS (
            SELECT 1
            FROM Tiki_Product
            WHERE Product_id = inserted.Product_id
        )
    )
    BEGIN
        RAISERROR('Error: Product does not exist.', 16, 1)
        ROLLBACK TRANSACTION
    END
END

```

Hình 39: Trigger kiểm tra tính hợp lệ

2.8 Trigger: Cập nhật tự động giá trị giảm giá khi giá sản phẩm thay đổi

```

--Cập nhật lại % giảm giá sau khi đổi giá sản phẩm
CREATE TRIGGER UpdateDiscountRateOnPriceChange
ON Tiki_Product
AFTER UPDATE
AS
BEGIN
    IF UPDATE (Price) OR UPDATE (Original_price)
    BEGIN
        UPDATE Tiki_Product
        SET
            Discount_rate = ROUND(((Original_price - Price) / Original_price) * 100, 0)
        WHERE Product_id = (SELECT Product_id FROM INSERTED);
    END;
END;

```

Hình 40: Trigger cập nhật tự động giá trị giảm giá

R4: Sao lưu và khôi phục dữ liệu

1. Xác định loại và thời gian backup dữ liệu

Backup dữ liệu là quá trình sao chép và lưu trữ bản sao của dữ liệu từ hệ thống vào thiết bị hoặc vị trí khác để đảm bảo dữ liệu quan trọng có thể khôi phục toàn vẹn khi có sự cố xảy ra như hư hỏng phần cứng, lỗi phần mềm,...

Có 3 loại backup dữ liệu phổ biến:

Loại Backup	Full Backup	Differential Backup	Transaction Log Backup
Định nghĩa	Sao lưu toàn bộ cơ sở dữ liệu	Sao lưu các thay đổi từ lần full backup gần nhất	Sao lưu các giao dịch ghi vào log từ lần sao lưu log gần nhất
Thời gian sao lưu	Mất nhiều thời gian nếu database lớn	Nhanh hơn full backup vì chỉ lưu những thay đổi	Nhanh nhất, chỉ sao lưu các giao dịch gần đây
Phục hồi	Nhanh, có thể phục hồi toàn bộ database	Chậm và cần phục hồi full backup trước	Chậm, cần phục hồi full backup và các log backup trước
Thời gian Backup	Thực hiện hàng tuần vào những ngày ít có thay đổi, thường là cuối tuần. Thời điểm ngoài giờ làm việc hoặc lúc hệ thống ít hoạt động.	Thực hiện hàng ngày hoặc vài lần trong ngày. Thời điểm cố định trong ngày hoặc sau các lần cập nhật quan trọng.	Thực hiện từ 5–15 phút một lần. Thực hiện thường xuyên trong ngày để hỗ trợ khôi phục tại thời điểm cụ thể.

Cú pháp	BACKUP DATABASE [Tên_CSDL] TO DISK = 'đường_dẫn\tên_file.bak'	BACKUP DATABASE [Tên_CSDL] TO DISK = 'đường_dẫn\tên_file.bak' WITH DIFFERENTIAL	BACKUP LOG [Tên_CSDL] TO DISK = 'đường_dẫn\tên_file.bak'
----------------	---------------------------------------------------------------	---------------------------------------------------------------------------------	----------------------------------------------------------

Nhóm sử dụng cả 3 loại backup trên vì với cơ sở dữ liệu về các sản phẩm, khách hàng (bao gồm ID, tên) và bình luận của khách về các sản phẩm đã mua, thì nhóm nhận thấy dữ liệu này không quá lớn, tuy nhiên có thể bị thay đổi nhiều (ở phần bình luận). Cụ thể nhóm thực hiện:

- Full Backup: thực hiện sao lưu lần đầu tiên và hàng tuần vào lúc 0 giờ ngày chủ nhật.
- Differential Backup: thực hiện sao lưu lần đầu tiên và hàng ngày vào 0 giờ mỗi tối
- Transaction Log Backup: thực hiện sao lưu lần đầu tiên và mỗi 15 phút hàng ngày

Cách 1: Sử dụng câu lệnh T-SQL:

- **Bước 1:** Tạo bản sao lưu đầu tiên:

```
BACKUP DATABASE DATA_For_ML
TO DISK = 'C:\DATA_Backup\Full_Backup.bak';

BACKUP DATABASE DATA_For_ML TO DISK = 'C:\DATA_Backup\Differential_Backup.bak'
WITH DIFFERENTIAL;

BACKUP LOG DATA_For_ML TO DISK = 'C:\DATA_Backup\Log_Backup.bak';
```

Hình 41: Câu lệnh T-SQL tạo sao lưu.

- **Bước 2:** Lập lịch tự động:

```

-- Lịch chạy Full Backup: Mỗi tuần một lần (chủ nhật 12 giờ sáng)
EXEC sp_add_jobschedule
    @job_name = N'Auto_Backup_DATA_For_ML',
    @name = N'Weekly Full Backup',
    @freq_type = 8, -- Chạy hàng tuần
    @freq_interval = 1, -- Chủ nhật
    @active_start_time = 000000; -- 12:00 tối
GO

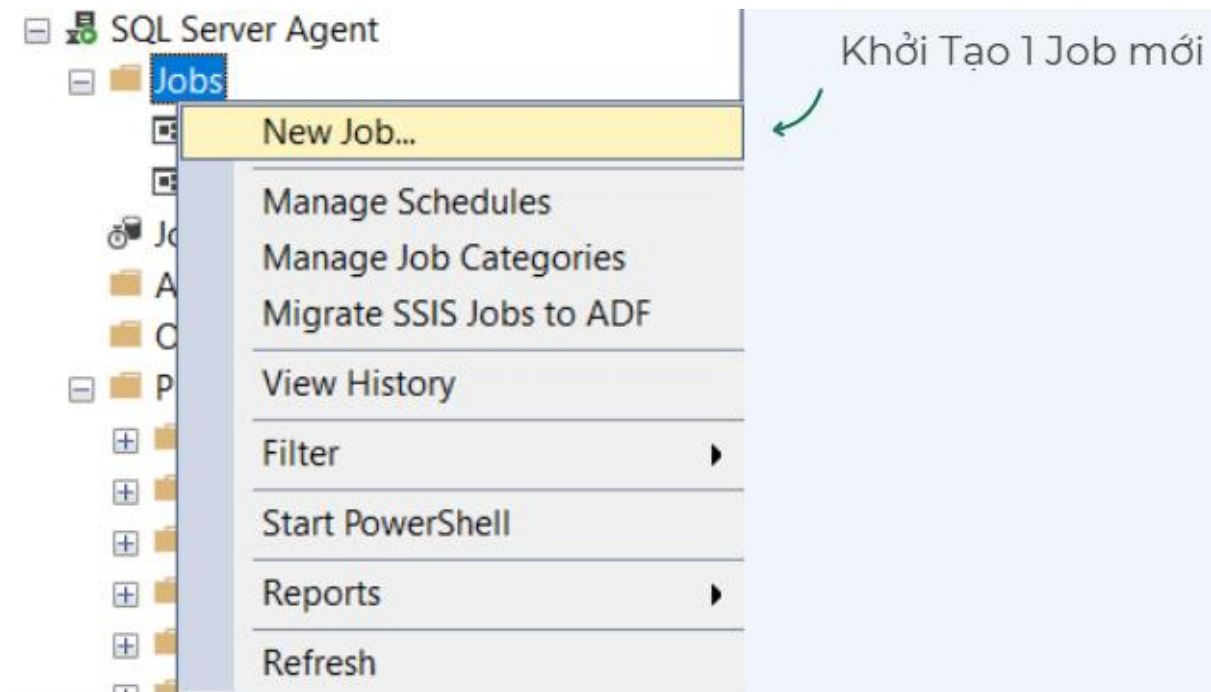
-- Lịch chạy Differential Backup: Mỗi ngày vào 12 giờ tối
EXEC sp_add_jobschedule
    @job_name = N'Auto_Backup_DATA_For_ML',
    @name = N'Daily Differential Backup',
    @freq_type = 4, -- Chạy hàng ngày
    @freq_interval = 1,
    @active_start_time = 000000; -- 12:00 tối
GO

-- Lịch chạy Log Backup: Mỗi 15 phút
EXEC sp_add_jobschedule
    @job_name = N'Auto_Backup_DATA_For_ML',
    @name = N'15-Minute Log Backup',
    @freq_type = 4, -- Lặp lại hàng ngày
    @freq_interval = 1, -- Chạy mỗi ngày
    @freq_subday_type = 4, -- Lặp lại theo phút
    @freq_subday_interval = 15, -- Mỗi 15 phút
    @active_start_time = 000000, -- Bắt đầu từ 12:00 sáng
    @active_end_time = 235959; -- Kết thúc lúc 11:59 tối
GO

```

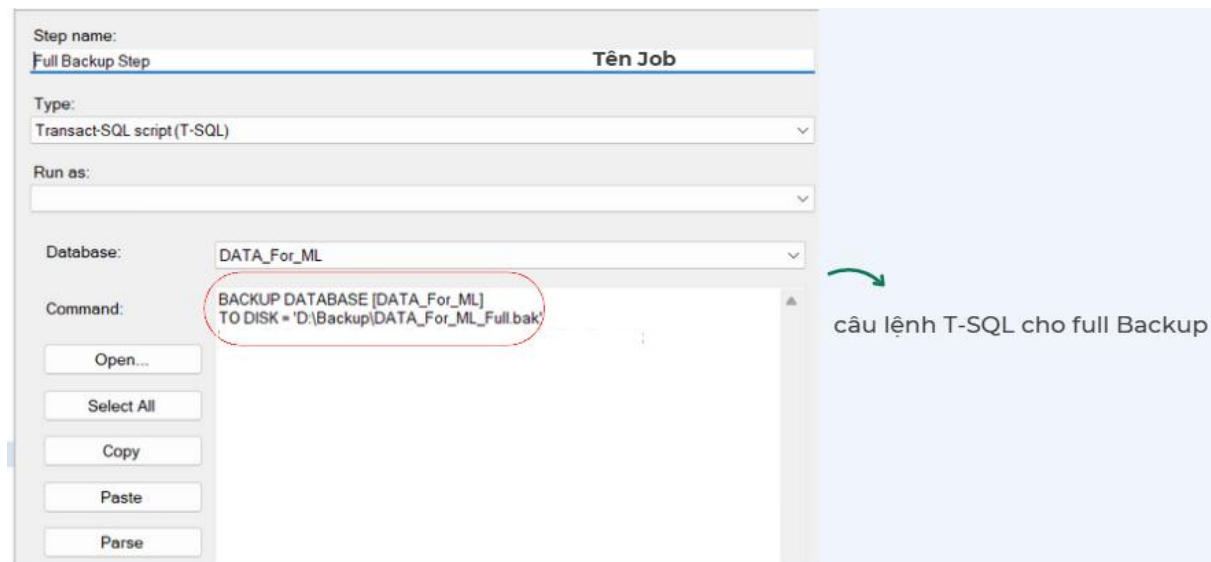
Hình 42: Câu lệnh T-SQL để lập lịch tự động tạo sao lưu.

Cách 2: Sử dụng công cụ:



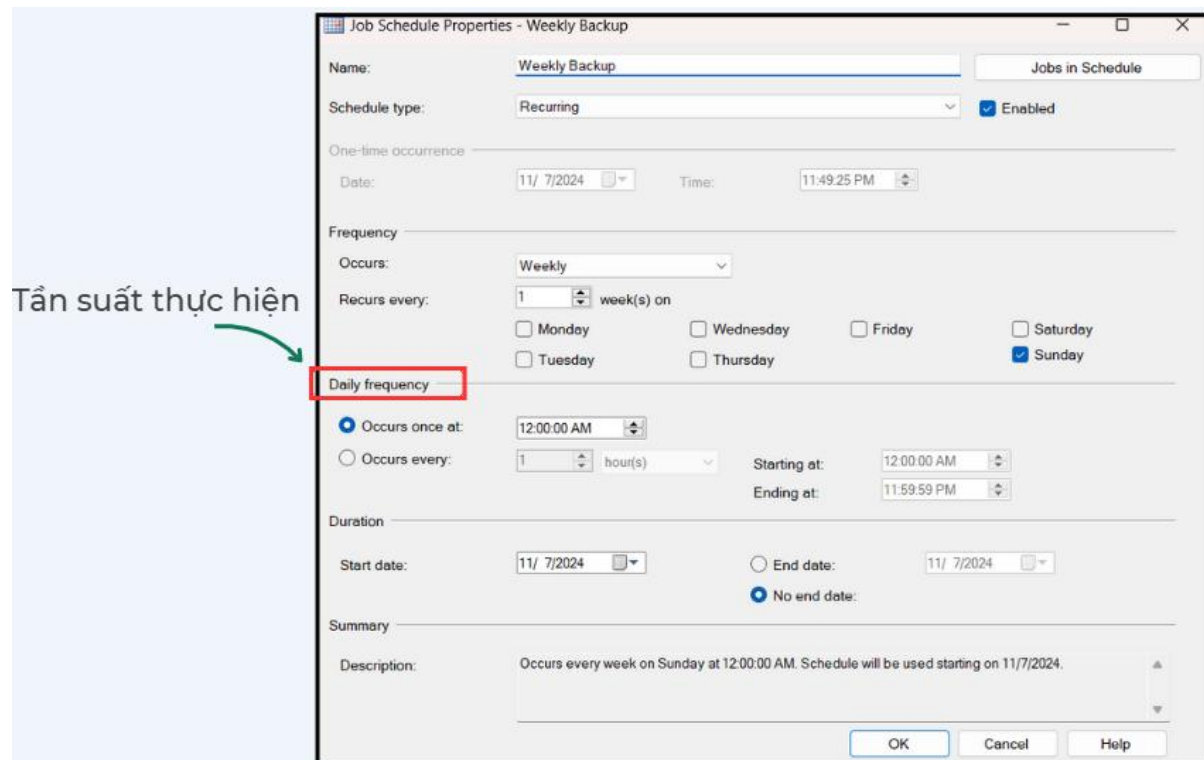
Hình 43: Cách sử dụng giao diện để tạo sao lưu.

- **Bước 1:** Thiết lập tên, loại và đường dẫn đích đến



Hình 44: Cách sử dụng giao diện để tạo sao lưu.

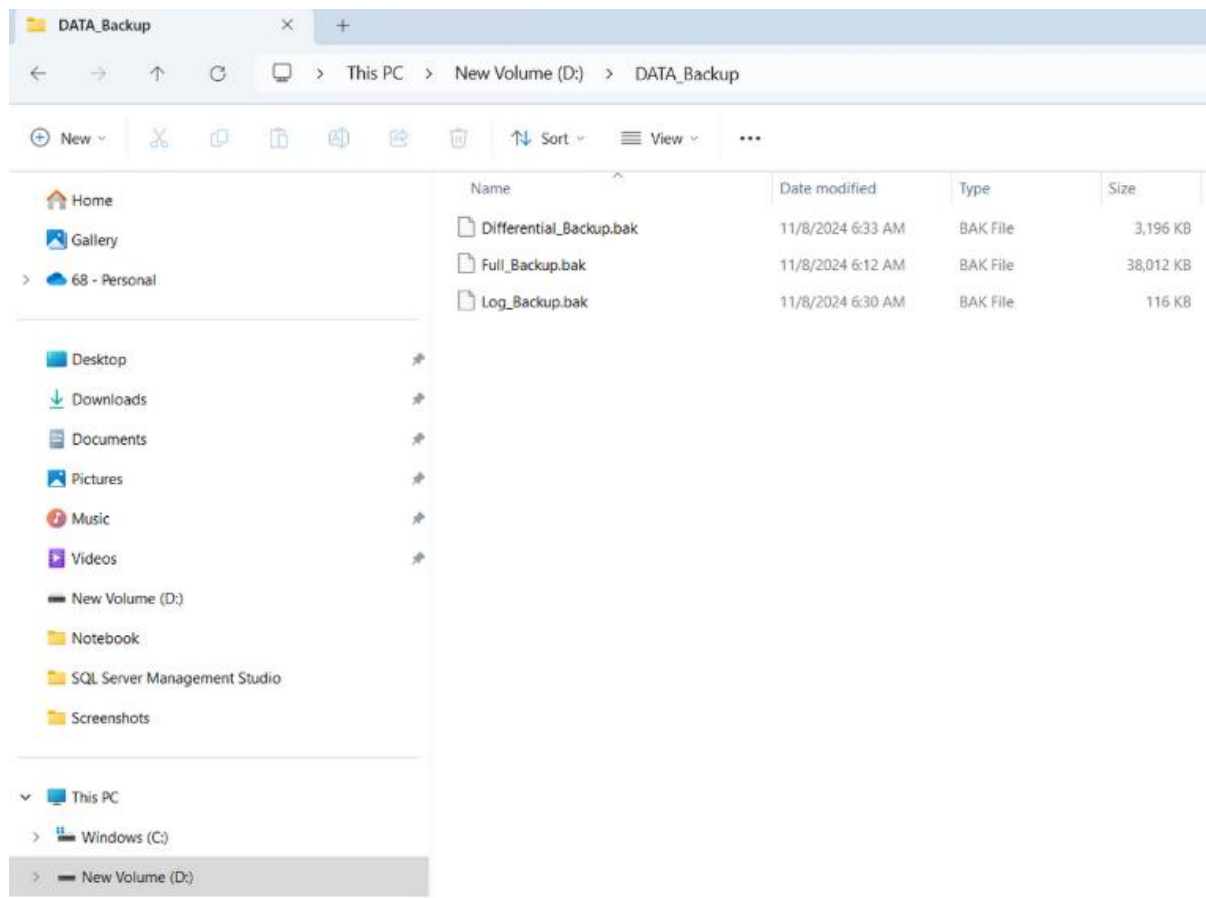
- **Bước 2:** Thiết lập thời gian cập nhật tự động cho Backup



Hình 45: Cách sử dụng giao diện để lập lịch tự động.

Thực hiện tương tự với 2 loại backup dữ liệu còn lại.

=> **Kết quả:**

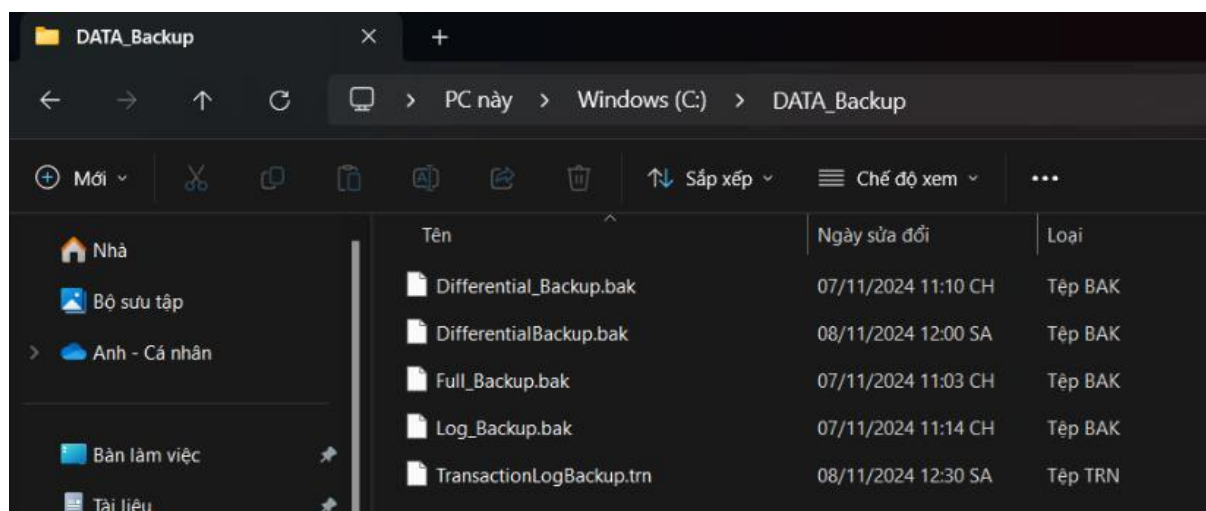


Hình 46: Các bản sao lưu đầu tiên.

2. Lưu trữ và quản lý backup

Nhóm đã thực hiện lưu trữ ở 2 nơi:

Lưu trữ cục bộ: Backup được lưu trực tiếp trên ổ cứng của máy chủ.

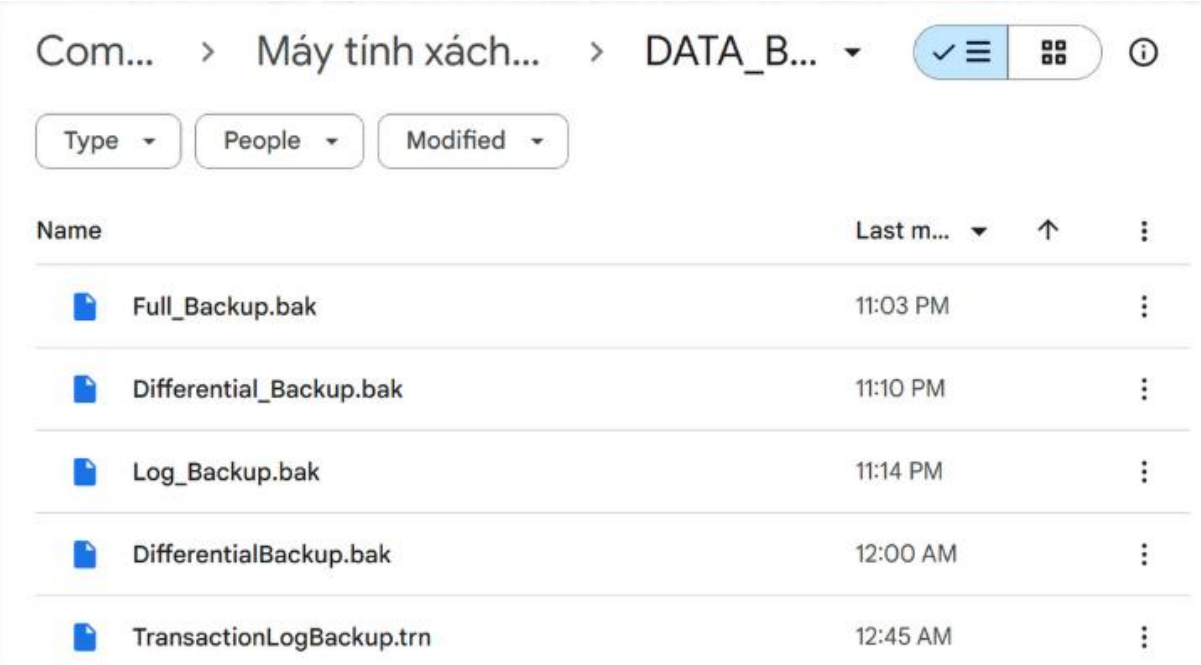


Hình 47: Nơi lưu trữ cục bộ.

- **Ưu điểm:**
 - + Tốc độ truy cập nhanh

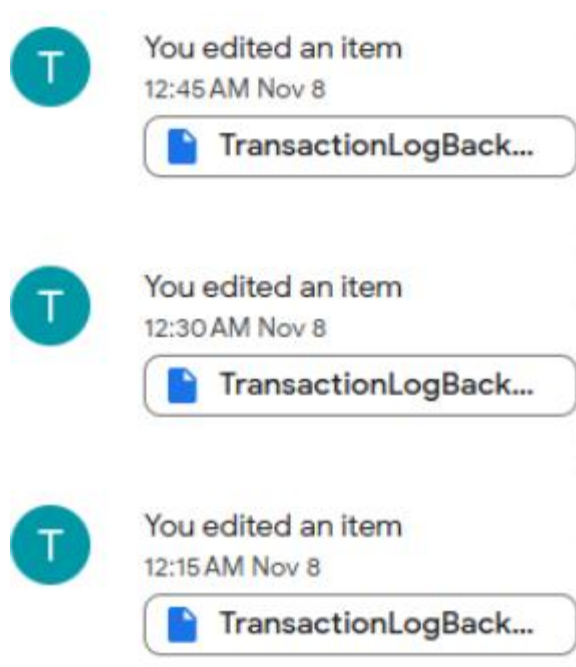
- + Thuận tiện cho việc khôi phục dữ liệu trong trường hợp xảy ra sự cố.
- **Nhược điểm:** Nếu máy chủ gặp sự cố như hư hỏng ổ đĩa, dữ liệu có thể bị mất hoàn toàn.

Lưu trữ đám mây trên Google Drive:



Name	Last m...	
Full_Backup.bak	11:03 PM	
Differential_Backup.bak	11:10 PM	
Log_Backup.bak	11:14 PM	
DifferentialBackup.bak	12:00 AM	
TransactionLogBackup.trn	12:45 AM	

Hình 48: Nơi lưu trữ đám mây.



Hình 49: Thông tin các bản sao lưu ở đám mây.

- **Ưu điểm:**
 - + Dễ thiết lập và truy cập từ bất kỳ đâu.
 - + Đảm bảo dữ liệu không bị mất nếu máy tính gặp sự cố.
- **Nhược điểm:**
 - + Hạn chế về dung lượng lưu trữ miễn phí.
 - + Bảo mật và tốc độ không tối ưu như các dịch vụ đám mây chuyên dụng.

Việc quản lý backup trong Microsoft SQL Server là một nhiệm vụ quan trọng nhằm đảm bảo dữ liệu của bạn được bảo vệ an toàn trước những sự cố bất ngờ.

2.1 Xác định chiến lược backup

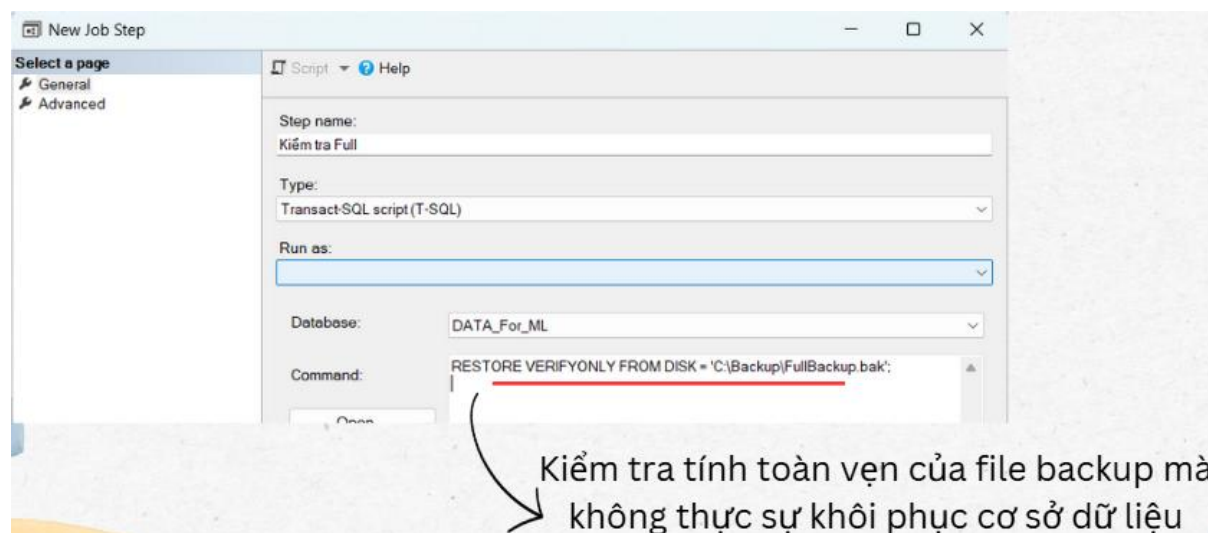
Trước khi bắt đầu quản lý backup, cần xác định loại và thời gian backup phù hợp với nhu cầu:

- Full Backup: thực hiện sao lưu lần đầu tiên và hàng tuần vào lúc 0 giờ ngày chủ nhật.
- Differential Backup: thực hiện sao lưu lần đầu tiên và hàng ngày vào 0 giờ mỗi tối
- Transaction Log Backup: thực hiện sao lưu lần đầu tiên và mỗi 15 phút hàng ngày
-

2.2 Kiểm tra tính toàn vẹn của backup

Sử dụng tùy chọn VERIFYONLY để kiểm tra tính toàn vẹn của file backup mà không cần khôi phục có sở dữ liệu:

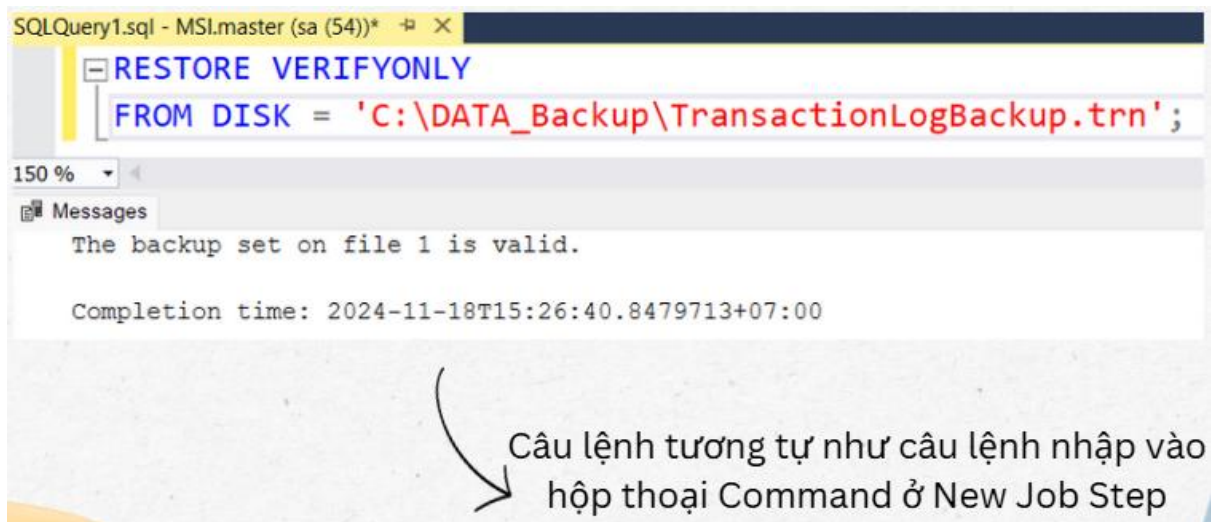
- **Cách 1:** Sử dụng công cụ:



Hình 50: Kiểm tra tính toàn vẹn của backup bằng giao diện

Sau đó, tạo job kiểm tra tính toàn vẹn của dữ liệu backup cho từng loại backup khác nhau. Cài lịch để tự động kiểm tra: full backup: tự động kiểm tra hàng tuần, differential backup: tự động kiểm tra hàng ngày, transaction log backup: tự động kiểm tra hàng ngày mỗi 30 phút.

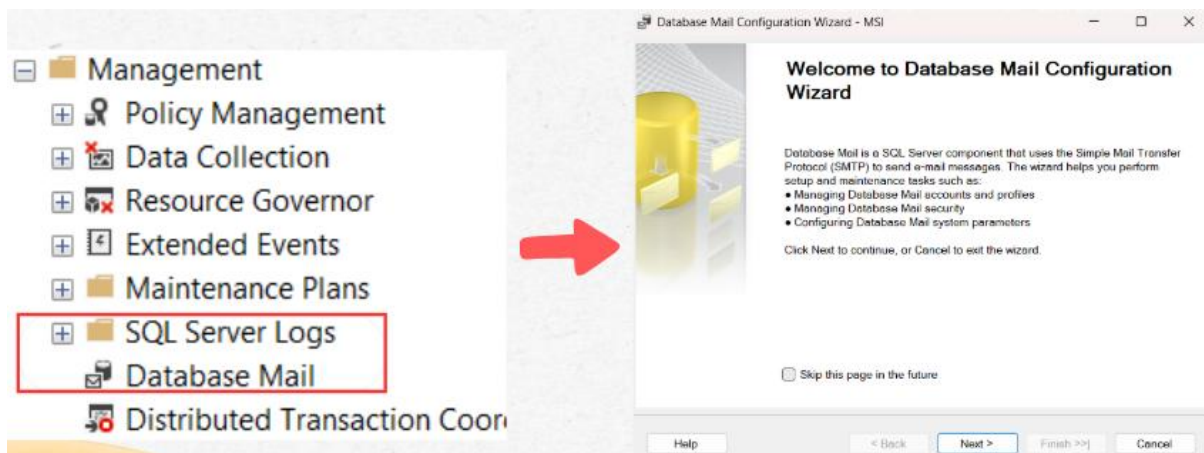
- **Cách 2:** Sử dụng câu lệnh T-SQL



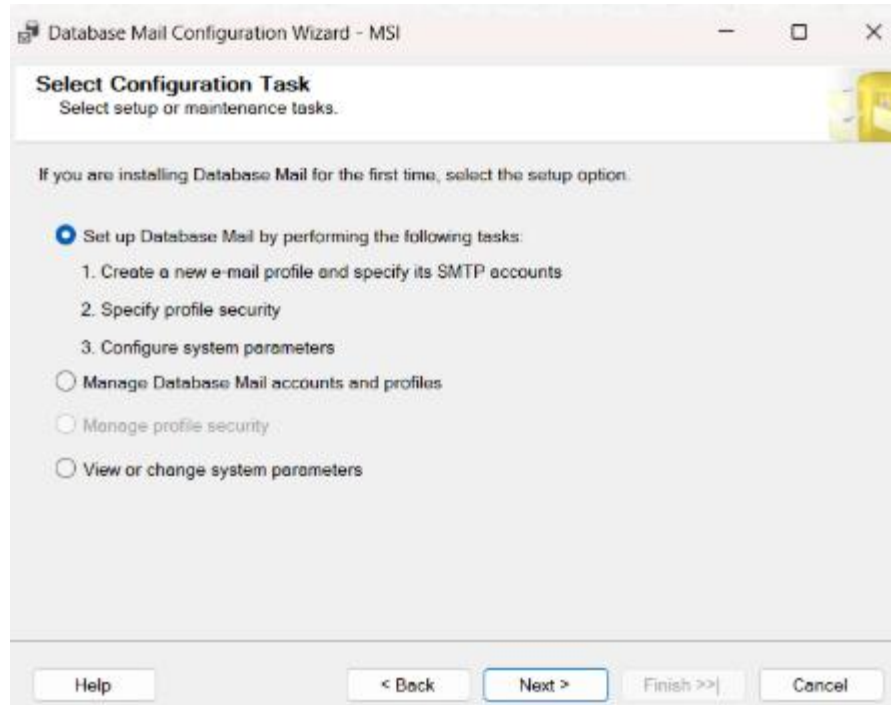
Hình 51: Kiểm tra tính toàn vẹn của backup bằng T-SQL

2.3 Giám sát backup qua Mail

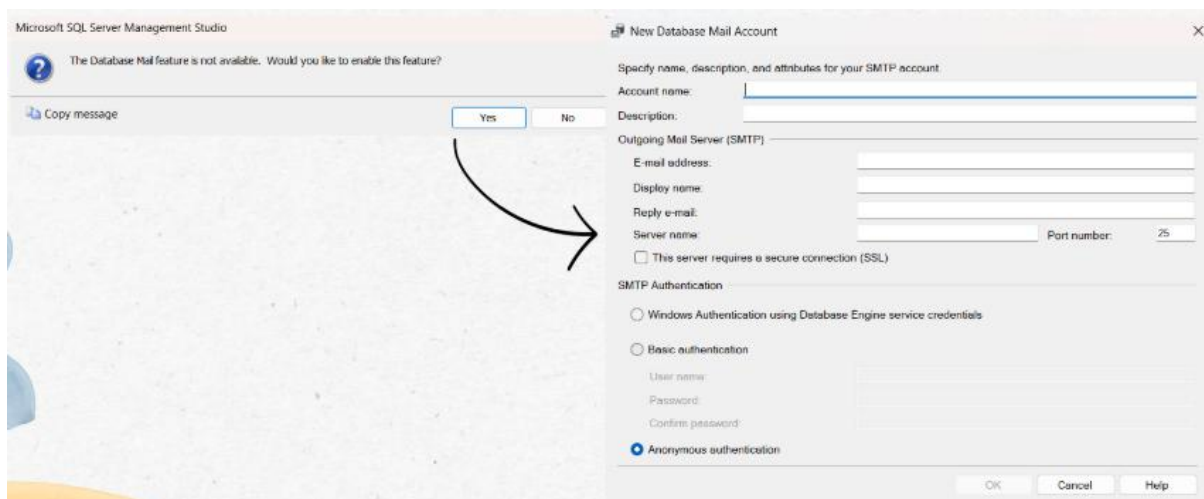
Cài đặt Database Mail



Hình 52: Cài đặt Mail.



Hình 53: Cài đặt Mail.



Hình 54: Cài đặt Mail.

Nhập các thông tin của Mail của bạn và tên, mô tả của mail dự định sẽ gửi để SQL có thể gửi cảnh báo đến.

Tạo một Job để gửi báo cáo hàng tuần

Step name: Weekly Backup Report

Type: Transact-SQL script (T-SQL)

Run as:

Database: DATA_For_ML

Command:

```

DECLARE @message NVARCHAR(MAX);
SET @message = "";

SELECT @message += 'Database: ' + name + CHAR(10) +
        'Last Full Backup: ' + ISNULL(CONVERT(VARCHAR, MAX(backup_fini
FROM msdb.dbo.backupset
WHERE type = 'D'
GROUP BY name;

EXEC msdb.dbo.sp_send_dbmail
    @profile_name = 'BackupAlerts',
    @recipients = 'phanxuanhaianh@gmail.com',
    @subject = 'Weekly Backup Report',
    @body = @message;

```

Buttons: Open..., Select All, Copy, Paste, Parse

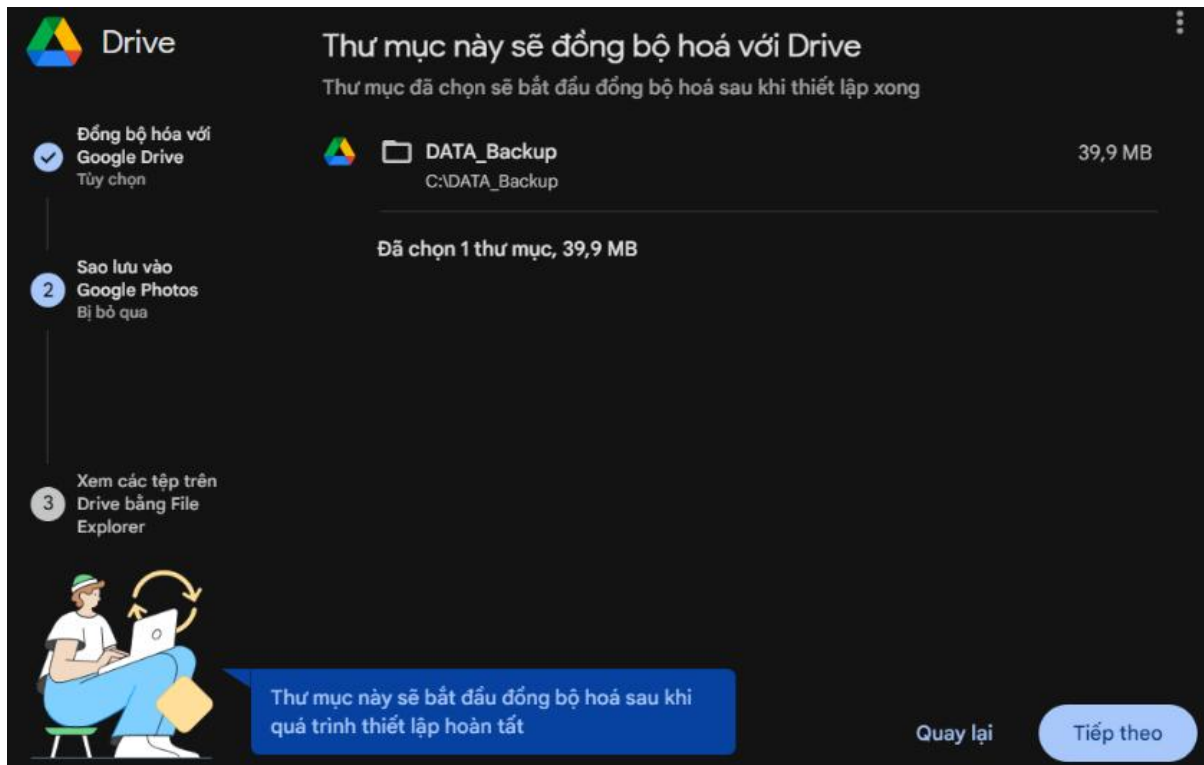
Hình 55: Giám sát backup qua Mail.

Khởi tạo biến để lưu thông tin báo cáo, truy vấn ngày hoàn thành backup gần nhất, rồi gửi báo cáo qua email.

Tương tự như cách lập lịch ở phần lập lịch để tự động sao lưu, nhóm cũng đã lập lịch cho Job này chạy vào thời gian định kỳ lúc 1 giờ sáng ngày chủ nhật hàng tuần. Nó sẽ tự động tạo báo cáo về các bản sao lưu và gửi báo cáo qua email đến người nhận đã chỉ định.

2.4 Sao lưu backup tới thiết bị ngoài

Nhóm đã cài đặt để các file backup có thể tự động được đồng bộ lên thư mục đám mây ở Google Drive để phòng ngừa khi có sự cố với máy tính, ổ đĩa hoặc cơ sở dữ liệu.



Hình 56: Sao lưu backup tới thiết bị ngoài.

2.5 Kiểm tra lịch sử backup

Truy vấn thông tin chi tiết về các bản sao lưu, bao gồm thời gian bắt đầu và kết thúc, loại sao lưu, và tên thiết bị lưu trữ, rồi sắp xếp theo thời gian sao lưu kết thúc.

```
SELECT
    database_name,
    backup_start_date,
    backup_finish_date,
    CASE
        WHEN type = 'D' THEN 'Full Backup'
        WHEN type = 'I' THEN 'Differential Backup'
        WHEN type = 'L' THEN 'Transaction Log Backup'
    END AS BackupType,
    physical_device_name
FROM msdb.dbo.backupset
INNER JOIN msdb.dbo.backupmediafamily
ON backupset.media_set_id = backupmediafamily.media_set_id
ORDER BY backup_finish_date DESC;
```

Hình 57: Kiểm tra lịch sử backup.

=> Kết quả truy vấn:

Câu lệnh trên sẽ trả về một bảng chứa các thông tin như:

- Tên cơ sở dữ liệu đã sao lưu.
- Thời gian bắt đầu và kết thúc sao lưu.
- Loại sao lưu đã thực hiện (Full, Differential, hoặc Log).
- Địa chỉ cụ thể của các file backup.

221	DATA_For_ML	2024-11-10 09:15:00.000	2024-11-10 09:15:00.000	Transaction Log Backup	C:\DATA_Backup\TransactionLogBackup.trn
222	DATA_For_ML	2024-11-10 09:12:09.000	2024-11-10 09:12:09.000	Transaction Log Backup	C:\DATA_Backup\TransactionLogBackup.trn
223	DATA_For_ML	2024-11-10 00:15:00.000	2024-11-10 00:15:00.000	Transaction Log Backup	C:\DATA_Backup\TransactionLogBackup.trn
224	DATA_For_ML	2024-11-10 00:00:01.000	2024-11-10 00:00:01.000	Transaction Log Backup	C:\DATA_Backup\TransactionLogBackup.trn
225	DATA_For_ML	2024-11-10 00:00:01.000	2024-11-10 00:00:01.000	Full Backup	C:\DATA_Backup\FullBackup.bak
226	DATA_For_ML	2024-11-10 00:00:01.000	2024-11-10 00:00:01.000	Differential Backup	C:\DATA_Backup\DifferentialBackup.bak
227	DATA_For_ML	2024-11-09 23:45:01.000	2024-11-09 23:45:01.000	Transaction Log Backup	C:\DATA_Backup\TransactionLogBackup.trn
228	DATA_For_ML	2024-11-09 23:30:00.000	2024-11-09 23:30:00.000	Transaction Log Backup	C:\DATA_Backup\TransactionLogBackup.trn
229	DATA_For_ML	2024-11-09 23:15:02.000	2024-11-09 23:15:02.000	Transaction Log Backup	C:\DATA_Backup\TransactionLogBackup.trn
230	DATA_For_ML	2024-11-09 23:00:00.000	2024-11-09 23:00:00.000	Transaction Log Backup	C:\DATA_Backup\TransactionLogBackup.trn
231	DATA_For_ML	2024-11-09 22:45:00.000	2024-11-09 22:45:00.000	Transaction Log Backup	C:\DATA_Backup\TransactionLogBackup.trn
232	DATA_For_ML	2024-11-09 22:30:00.000	2024-11-09 22:30:00.000	Transaction Log Backup	C:\DATA_Backup\TransactionLogBackup.trn
233	DATA_For_ML	2024-11-09 22:15:00.000	2024-11-09 22:15:01.000	Transaction Log Backup	C:\DATA_Backup\TransactionLogBackup.trn

Hình 58: Lịch sử backup.

3. Phục hồi dữ liệu

Thực hiện hồi phục dữ liệu khi:

- Xảy ra sự cố hệ thống dẫn đến mất mát dữ liệu, chẳng hạn như lỗi phần cứng (ổ cứng hỏng), lỗi phần mềm hoặc sự cố máy chủ
- Có sự cố do thao tác sai, chẳng hạn như lệnh DELETE hoặc UPDATE không mong muốn trên một bảng dữ liệu quan trọng.

Khi đó, việc phục hồi bản sao lưu cho phép khôi phục dữ liệu về trạng thái trước khi sự cố xảy ra hoặc giúp lấy lại dữ liệu bị thay đổi hoặc xóa nhầm.

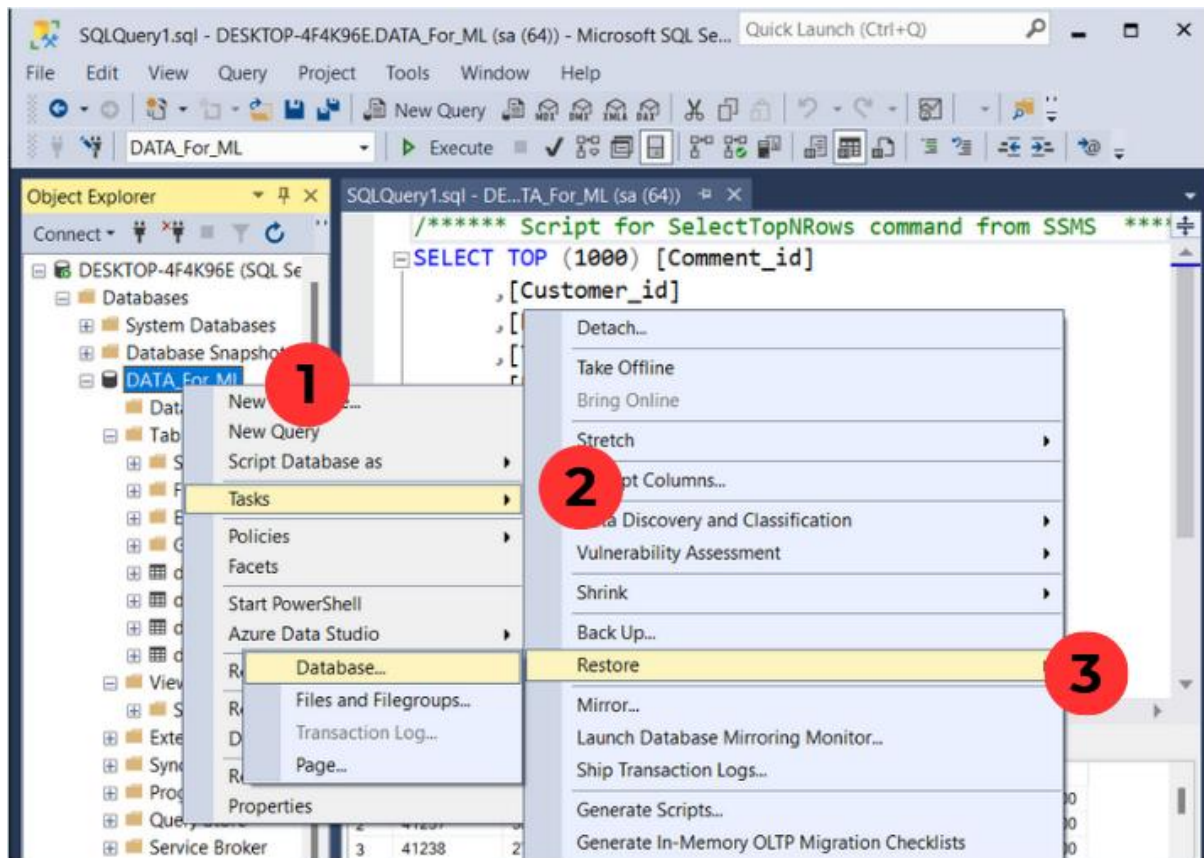
Bước 1: Đảm bảo không có ai đang truy cập vào cơ sở dữ liệu

Trước khi phục hồi, hãy đảm bảo rằng không có người dùng nào đang truy cập vào cơ sở dữ liệu để tránh lỗi khi phục hồi. Có thể thực hiện lệnh sau để đặt cơ sở dữ liệu ở chế độ đơn người dùng:

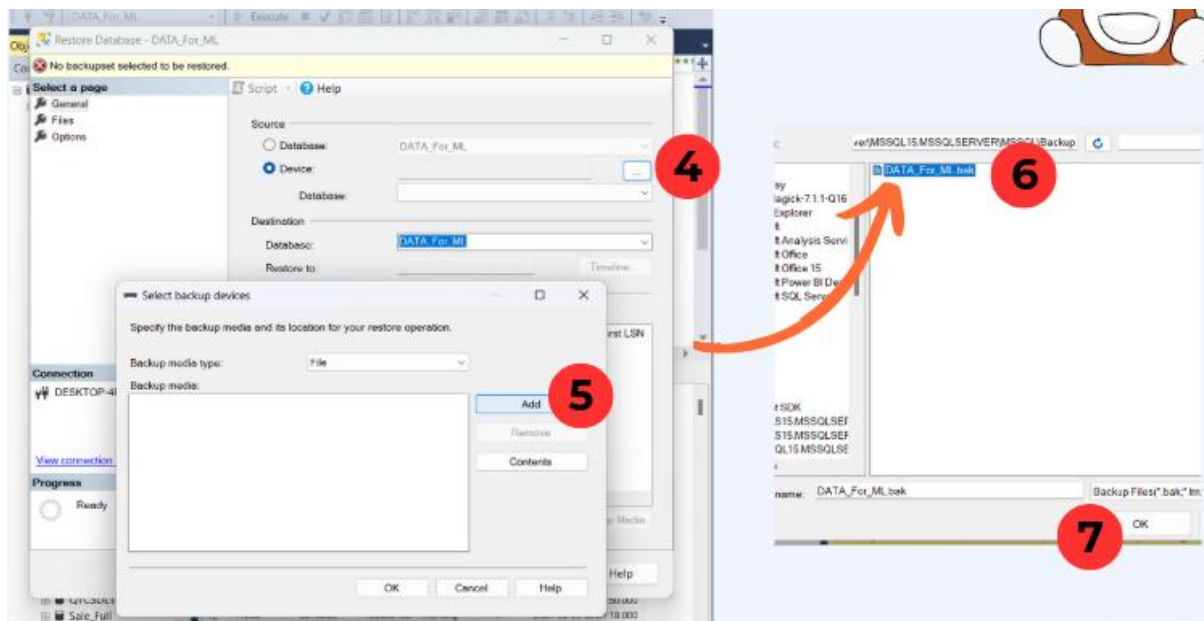
```
ALTER DATABASE [DATA_For_ML] SET SINGLE_USER WITH ROLLBACK IMMEDIATE;
```

Bước 2: Tiến hành phục hồi từ bản sao lưu

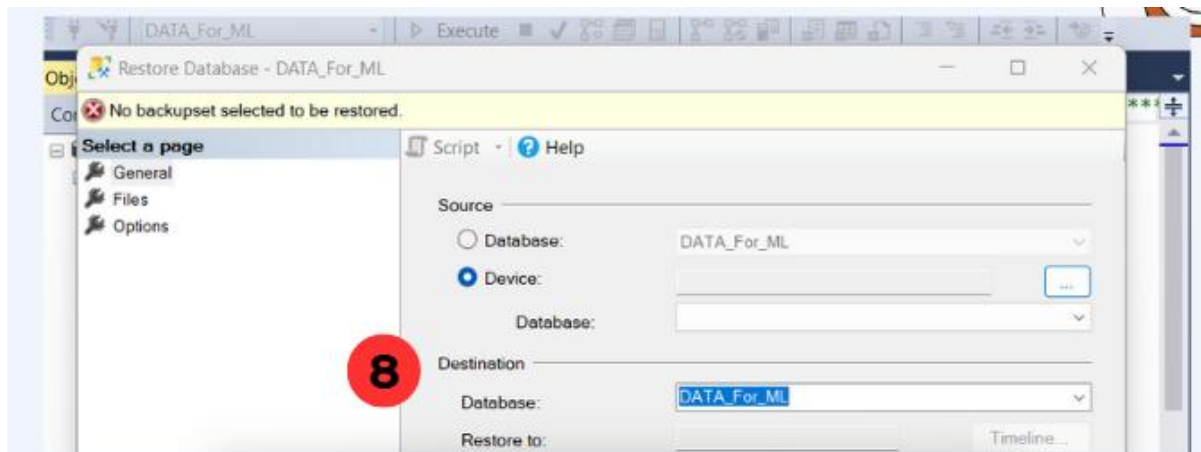
- **Cách 1:** Sử dụng công cụ



Hình 59: Các bước phục hồi dữ liệu.



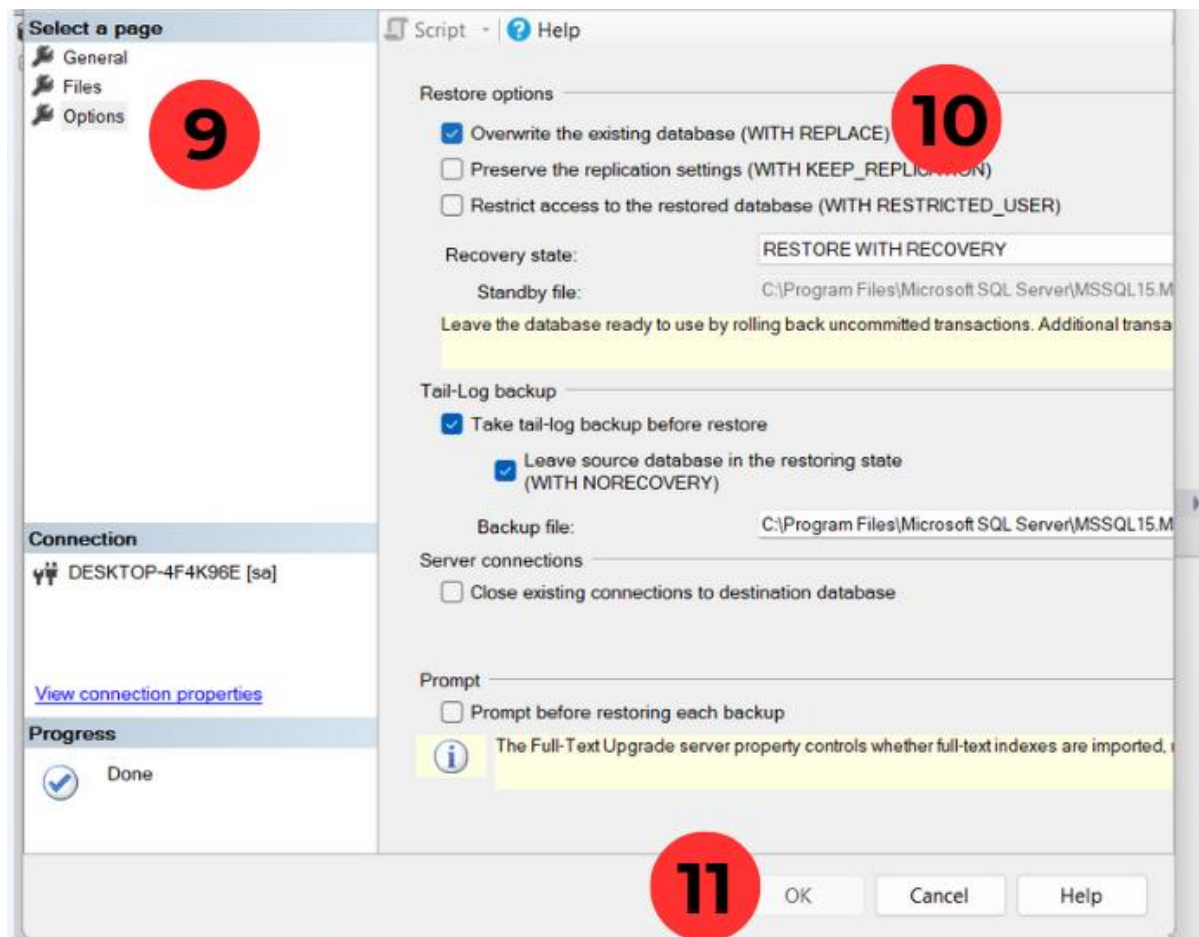
Hình 60: Các bước phục hồi dữ liệu.



Destination: Chọn CSDL đích để phục hồi hoặc điền tên mới nếu muốn tạo một cơ sở dữ liệu mới từ bản sao lưu.

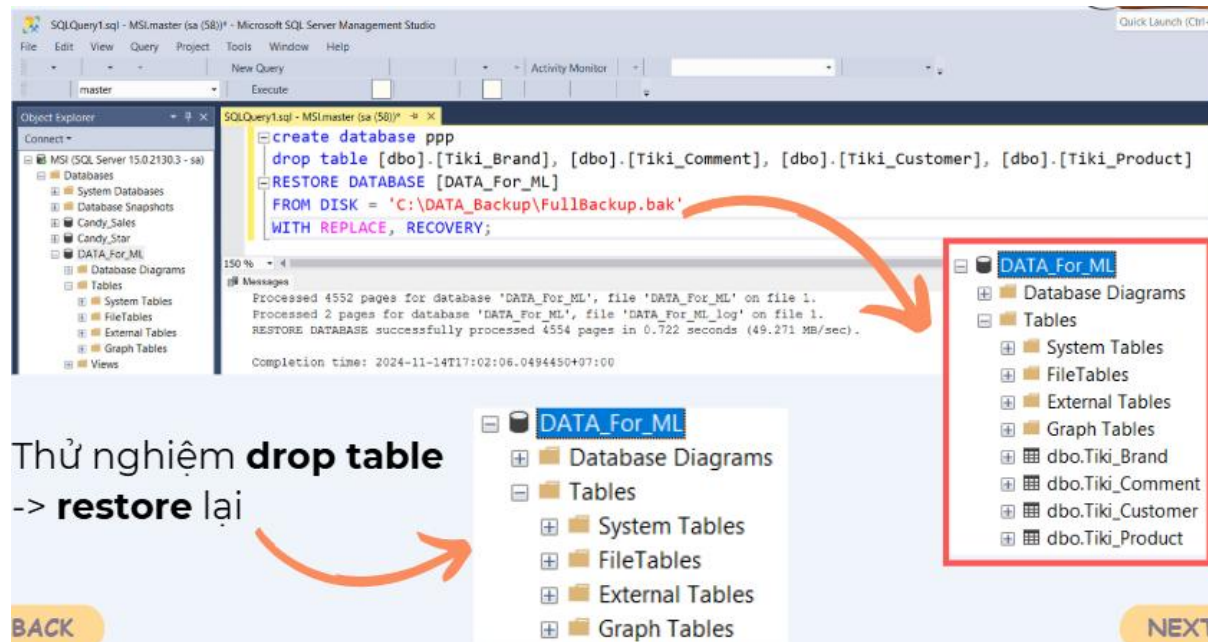
Hình 15:

Hình 61: Các bước phục hồi dữ liệu.



Hình 62: Các bước phục hồi dữ liệu.

- **Cách 2:** Sử dụng câu lệnh T-SQL:



Hình 63: Các bước phục hồi dữ liệu.

Bước 3: Đặt cơ sở dữ liệu về chế độ nhiều người dùng

Sau khi phục hồi hoàn tất, đưa cơ sở dữ liệu trở về chế độ nhiều người dùng:

```
ALTER DATABASE [DATA_For_ML] SET MULTI_USER;
```

Bước 4: Thực hiện các truy vấn kiểm tra

Sau khi hoàn thành phục hồi, có thể chạy một vài lệnh truy vấn kiểm tra dữ liệu để đảm bảo rằng quá trình phục hồi đã thành công, ví dụ:

```
USE [DATA_For_ML];
SELECT TOP 10 * FROM [Tiki_Product];
```

Lưu ý:

- Đảm bảo rằng có quyền truy cập và các quyền cần thiết để thực hiện phục hồi.
- Nếu phục hồi cơ sở dữ liệu lớn, bạn nên thực hiện trong thời gian ít người sử dụng hệ thống để tránh ảnh hưởng đến các người dùng khác.

R5: Phân quyền trên cơ sở dữ liệu

Phân quyền trên cơ sở dữ liệu là một bước quan trọng nhằm đảm bảo an toàn và tính toàn vẹn của hệ thống. Khi áp dụng phân quyền, mỗi người dùng hoặc nhóm người dùng chỉ được phép truy cập và thao tác trên những dữ liệu hoặc chức năng phù hợp với vai trò của họ. Điều này giúp giảm thiểu nguy cơ rò rỉ, mất mát dữ liệu, đồng thời hạn chế quyền truy cập không cần thiết vào thông tin nhạy cảm. Bên cạnh đó, phân quyền còn hỗ trợ quản lý và vận hành hệ thống dễ dàng hơn, giúp xác định trách nhiệm khi có vấn đề xảy ra và tối ưu hóa tài nguyên. Do đó, nhóm tiến hành việc phân quyền trên cơ sở dữ liệu

1. Ma trận phân quyền:

Quyền hạn	Admin	DE (Data Engineer)	DA (Data Analyst)
Thêm dữ liệu (INSERT)			✗
Sửa dữ liệu (UPDATE)			✗
Xóa dữ liệu (DELETE)			✗
Truy vấn dữ liệu (SELECT)			
Truy cập VIEW			
Tạo/sửa/xóa VIEW			✗
Truy cập lịch sử/đã phân tích			
Quản lý thủ tục/hàm			✗
Toàn quyền (FULL CONTROL)		✗	✗

2. Phân quyền trên cơ sở dữ liệu:

Sau khi xác định được quyền hạn của từng nhóm người dùng là Admin, Data Engineer (DE) và Data Analyst (DA), nhóm tiến hành phân quyền trên cơ sở dữ liệu.

Đầu tiên là tạo Login cho từng nhóm người dùng, việc này nhằm mục đích xác thực người dùng khi họ truy cập vào hệ thống:

```

-- Login Admin
CREATE LOGIN Admin_Login
WITH PASSWORD = 'Admin@123',
    CHECK_POLICY = OFF,
    CHECK_EXPIRATION = OFF,
    DEFAULT_DATABASE = DATA_For_ML;
GO

-- Login DE
CREATE LOGIN DE_Login
WITH PASSWORD = 'DE@123',
    CHECK_POLICY = OFF,
    CHECK_EXPIRATION = OFF,
    DEFAULT_DATABASE = DATA_For_ML;
GO

```

Hình 15: Tạo Login cho người dùng.

```

-- Login DA
CREATE LOGIN DA_Login
WITH PASSWORD = 'DA@123',
    CHECK_POLICY = OFF,
    CHECK_EXPIRATION = OFF,
    DEFAULT_DATABASE = DATA_For_ML;
GO

```

Hình 64: Tạo Login cho người dùng.

Sau đó, nhóm tạo user cho từng login trong cơ sở dữ liệu, mục đích là để gắn login với user cụ thể để sử dụng quyền

```

-- Tạo user cho từng login trong cơ sở dữ liệu DATA
CREATE USER Admin_User FOR LOGIN Admin_Login;
GO

CREATE USER DE_User FOR LOGIN DE_Login;
GO

CREATE USER DA_User FOR LOGIN DA_Login;
GO

```

Hình 65: Tạo User cho từng Login.

Bước tiếp theo, nhóm tiến hành tạo vai trò (role) cho từng nhóm người dùng, đồng thời gán quyền cho từng role để đảm bảo rằng chỉ những người có vai trò tương ứng mới được thực hiện các chức năng như đã liệt kê ở ma trận phân quyền:

```

-- Tạo role Admin và cấp quyền toàn bộ cơ sở dữ liệu cho role này
CREATE ROLE Admin;
GO
GRANT CONTROL ON DATABASE::DATA_For_ML TO Admin;
GO

-- Tạo role DE
CREATE ROLE DE;
GO

GRANT SELECT, INSERT, UPDATE, DELETE ON DATABASE::DATA_For_ML TO DE; -- Cấp quyền thêm, sửa, xóa, truy vấn dữ liệu
GRANT EXECUTE ON SCHEMA::dbo TO DE; -- Cấp quyền thực thi các thủ tục và hàm
GRANT CREATE PROCEDURE TO DE; -- Cấp quyền tạo mới thủ tục
GRANT CREATE FUNCTION TO DE; -- Cấp quyền tạo mới hàm
GRANT ALTER ON SCHEMA::dbo TO DE; -- Cấp quyền cho phép sửa các thủ tục và hàm
GRANT CONTROL ON SCHEMA::dbo TO DE; -- Cấp quyền cho phép xóa các thủ tục và hàm
GO

```

```
-- Tạo role DA và cấp quyền chỉ truy vấn dữ liệu
CREATE ROLE DA;
GO
GRANT SELECT ON DATABASE::DATA_For_ML TO DA;
GO
```

Hình 66: Tạo Role và phân quyền cho từng Role.

Bước cuối cùng, nhóm tiến hành liên kết user với vai trò tương ứng để áp dụng quyền được phân:

```
-- Thêm các user vào role
ALTER ROLE Admin ADD MEMBER Admin_User;
GO

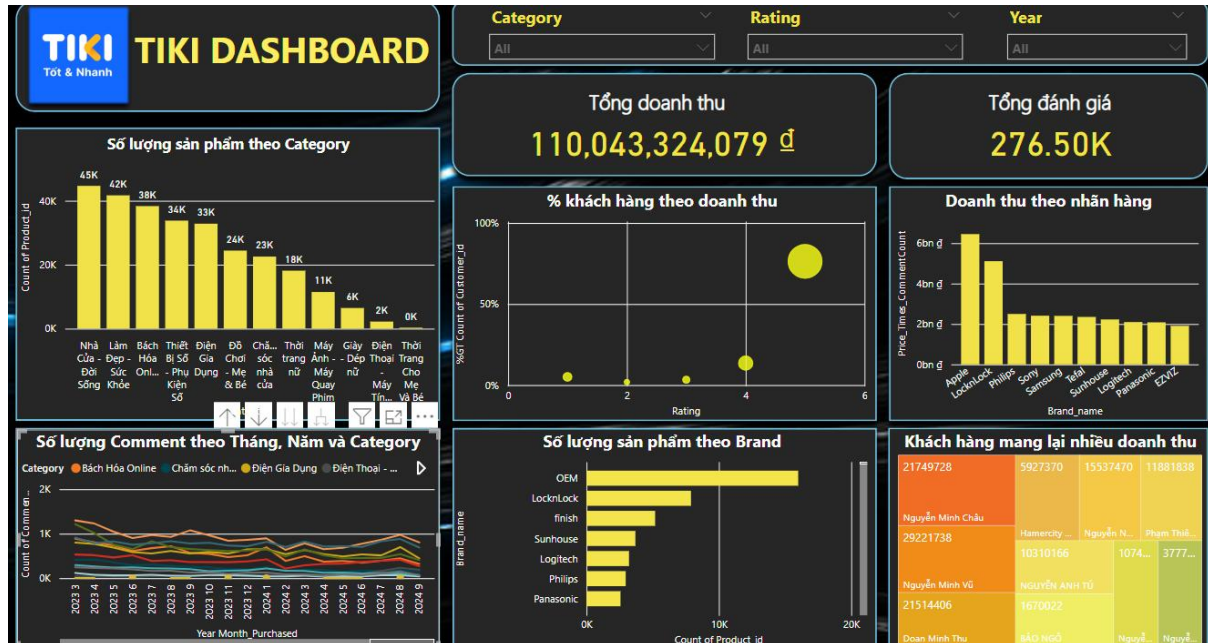
ALTER ROLE DE ADD MEMBER DE_User;
GO

ALTER ROLE DA ADD MEMBER DA_User;
GO
```

Hình 67: Thêm User vào Role.

R6: Trực quan hóa dữ liệu

1. Sử dụng Power BI



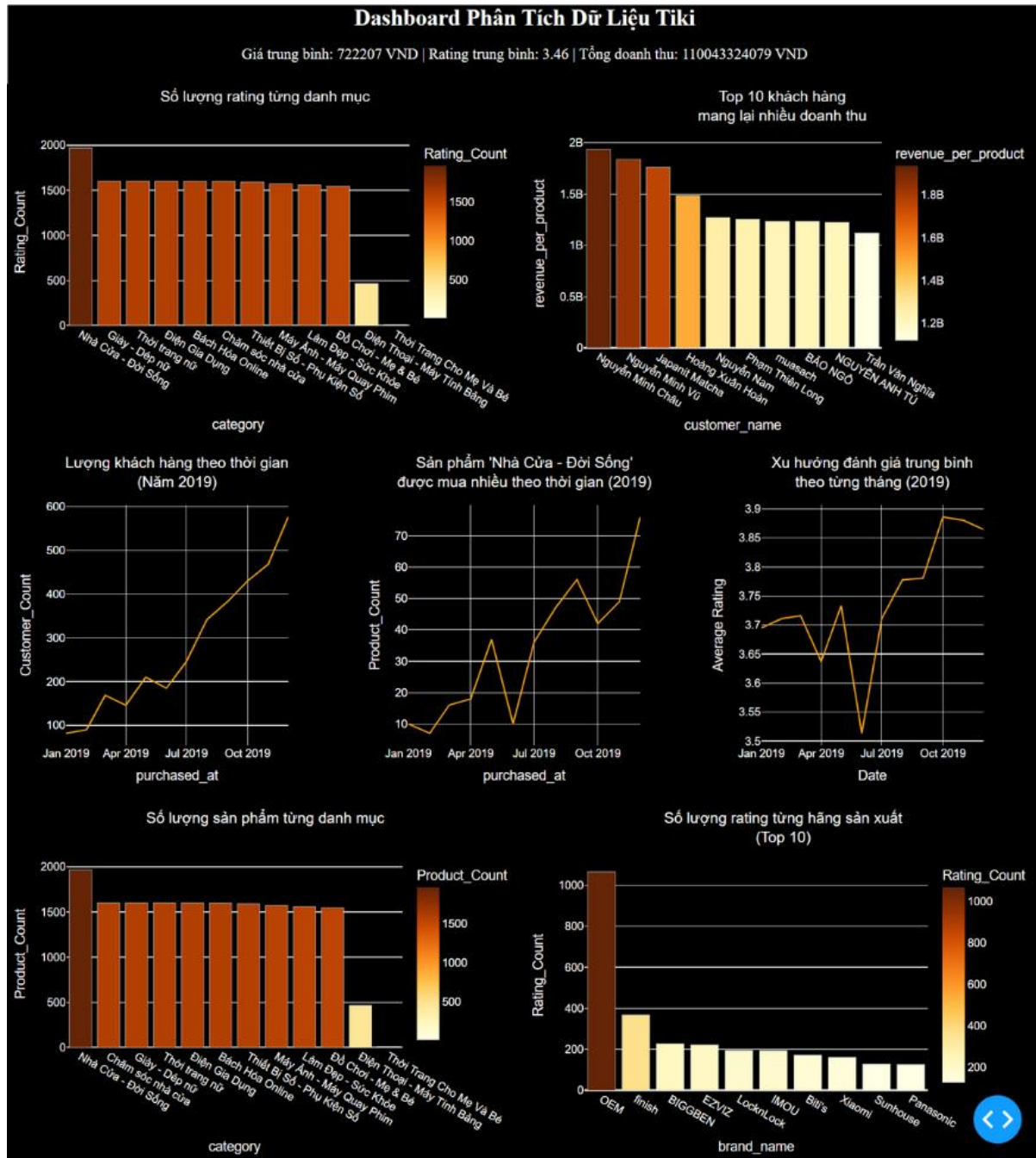
Hình 68: Dashboard bằng Power BI.

2. Sử dụng Tableau



Hình 69: Dashboard bằng Tableau.

3. Sử dụng Python



Hình 70: Dashboard bằng Python.

4. So sánh 3 công cụ sử dụng để trực quan hóa

Tiêu chí	PowerBI	Tableau	Python
Độ phổ biến	Cao trong lĩnh vực doanh nghiệp, đặc biệt là môi trường Microsoft.	Cao trong lĩnh vực doanh nghiệp, đặc biệt cho phân tích dữ liệu chuyên sâu và trực quan hóa đẹp mắt.	Rất phổ biến trong cộng đồng khoa học dữ liệu và lập trình.
Chi phí	Miễn phí (Power BI Desktop) hoặc tính phí (Power BI Pro/ Premium).	Trả phí (Tableau Public miễn phí nhưng hạn chế tính năng).	Miễn phí (Python là mã nguồn mở).
Dễ sử dụng	Dễ học, giao diện kéo-thả thân thiện với người dùng không chuyên lập trình.	Dễ sử dụng với giao diện trực quan, nhưng cần thời gian làm quen để khai thác hết tiềm năng.	Đòi hỏi kiến thức lập trình, khó hơn với người mới bắt đầu.
Khả năng kết nối dữ liệu	Hỗ trợ kết nối tốt với các sản phẩm của Microsoft và nhiều nguồn dữ liệu khác.	Hỗ trợ nhiều nguồn dữ liệu, đặc biệt mạnh trong phân tích dữ liệu lớn.	Kết nối đa dạng với thư viện như Pandas, SQLAlchemy, hoặc APIs.
Khả năng phân tích	Tốt cho phân tích cơ bản và tích hợp DAX cho các phép tính nâng cao.	Rất mạnh trong phân tích dữ liệu phức tạp và tạo bảng điều khiển tương tác.	Mạnh mẽ và linh hoạt cho phân tích từ cơ bản đến nâng cao với NumPy, Pandas, Scikit-learn, v.v.
Độ tùy chỉnh	Giới hạn ở các tính năng do Microsoft cung cấp.	Tùy chỉnh nhiều hơn Power BI nhưng vẫn bị hạn chế trong các tình huống phức tạp.	Tùy chỉnh tối đa, có thể làm bất kỳ loại đồ thị nào nếu lập trình được.
Thư viện biểu đồ	Hỗ trợ nhiều loại biểu đồ cơ bản và một số biểu đồ nâng cao.	Hỗ trợ nhiều loại biểu đồ, được tối ưu hóa về thẩm mỹ và tính tương tác.	Vô cùng đa dạng (Matplotlib, Seaborn, Plotly, Bokeh, v.v.)

Môi trường sử dụng	Chủ yếu dành cho doanh nghiệp và người dùng Microsoft Office.	Doanh nghiệp và phân tích chuyên sâu, thích hợp với các tổ chức chuyên nghiệp.	Lý tưởng cho nhà khoa học dữ liệu, lập trình viên, và các ứng dụng học thuật hoặc doanh nghiệp.
--------------------	---------------------------------------------------------------	--------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------

Tổng kết

- **Power BI:** Phù hợp với người dùng trong doanh nghiệp, đặc biệt là người dùng Microsoft.
- **Tableau:** Thích hợp để tạo báo cáo trực quan đẹp mắt và phân tích phức tạp.
- **Python:** Mạnh mẽ và linh hoạt nhất nhưng yêu cầu kỹ năng lập trình, phù hợp với các nhà phân tích dữ liệu chuyên sâu.

R7: Xây dựng mô hình đề xuất sản phẩm

1. Giới Thiệu

Trong giai đoạn này, nhóm thực hiện việc xây dựng một mô hình gợi ý sản phẩm tự động sử dụng ngôn ngữ Python và kỹ thuật machine learning dùng là KNN. Mục tiêu chính là tối ưu hóa việc gợi ý sản phẩm cho người dùng, giúp hệ thống nhanh chóng tìm kiếm và đề xuất những sản phẩm phù hợp nhất.

Bộ dữ liệu đã có được trích xuất từ SQL Server, bao gồm các thuộc tính như tên sản phẩm, giá bán, và các đặc điểm phụ. Giai đoạn này tập trung vào quá trình xử lý dữ liệu, xây dựng mô hình huấn luyện, và tạo một demo về trực quan hệ thống gợi ý sản phẩm khi nhập vào cũng như đưa ra 5 sản phẩm gợi ý đầu ra tương ứng.

2. Quá trình hoạt động của mô hình

2.1 Kết Nối Cơ Sở Dữ Liệu

Nhóm sử dụng thư viện pyodbc để kết nối với cơ sở dữ liệu SQL Server. Bảng dữ liệu Tiki Product được trích xuất bao gồm các thông tin quan trọng như tên sản phẩm, giá bán, và danh mục..

2.2 Xử Lý Dữ Liệu

Dữ liệu ban đầu được kết hợp thành một cột đặc trưng duy nhất bằng cách kết nối tên sản phẩm và giá. Việc tạo đặc trưng giúp tăng khả năng nhận diện tương đồng giữa các sản phẩm khi huấn luyện mô hình.

2.3 Vector Hóa Đặc Trưng

Nhóm sử dụng TfidfVectorizer để chuyển đổi các chuỗi đặc trưng thành vector số học. Quá trình này biến dữ liệu text thành dạng ma trận số học, từ đó sẽ tách biệt và có thể tính toán được số lượng của các đặc trưng của từ khóa trong dữ liệu.

2.4 Thuật Toán KNN (K-Nearest Neighbors)

KNN (K-Nearest Neighbors) là một thuật toán học máy phi tuyến tính, dựa trên nguyên tắc tìm k láng giềng gần nhất trong không gian đa chiều. KNN không có giai đoạn huấn luyện phức tạp, thay vào đó mỗi dữ liệu đầu vào được tính toán trực tiếp với dữ liệu huấn luyện đã có.

Quy Trình Hoạt Động:

- Bước 1: Chuyển dữ liệu vào dạng vector số học (bằng TF-IDF).
- Bước 2: Tính khoảng cách giữa vector đầu vào và tất các vector trong tập dữ liệu. Thường sử dụng khoảng cách Cosine hoặc Euclidean.
- Bước 3: Tìm ra k vector gần nhất (láng giềng gần nhất).

- Bước 4: Trả kết quả là danh sách láng giềng gần nhất này.

2.5 Tạo Giao Diện Tương Tác

Giao diện được xây dựng bằng thư viện gradio, cho phép người dùng nhập tên sản phẩm và nhận lại kết quả gợi ý. Giao diện trực quan giúp người dùng tương tác và kiểm tra kết quả một cách dễ dàng.

3. Mã Python

```
import gradio as gr
import pyodbc
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
from fuzzywuzzy import fuzz, process
from sklearn.neighbors import NearestNeighbors

server = 'DESKTOP-77HM0LQ'
database = 'DATA_For_ML'
username = 'sa'
password = 'taolakhaiden0285'

connection_String = f'DRIVER={{SQL
Server}};SERVER={server};DATABASE={database};UID={username};PWD={pa
ssword}'

try:
    # Kết nối với databse và in kết quả 5 sản phẩm đầu tiên của bảng Product
    conn = pyodbc.connect(connection_String)
    query = 'SELECT * FROM Tiki_Product'

    df_sanpham = pd.read_sql(query, conn)
    print(df_sanpham.head(5))

except pyodbc.Error as e:
    print(f'Error: {e}')
finally:
    if conn:
        conn.close()

# Tạo đặc trưng cho dữ liệu :
features = ['Product_name', 'Price']
```



```

def combineFeatures(row):
    return str(row['Price']) + " " + str(row['Product_name'])

df_sanpham['combinedFeatures'] = df_sanpham.apply(combineFeatures, axis=1)
print(df_sanpham['combinedFeatures'].head())

tf = TfidfVectorizer()
tfMatrix = tf.fit_transform(df_sanpham['combinedFeatures'])

similar = cosine_similarity(tfMatrix)
number = 5

def get_similar_products(Product_name):
    ketqua = []
    # Tìm kiếm sản phẩm gần giống
    Product_name_list = df_sanpham['Product_name'].tolist()
    best_match = process.extractOne(Product_name, Product_name_list)
    if best_match:
        Product_name = best_match[0]
        product_id = df_sanpham[df_sanpham['Product_name'] ==
Product_name]['Product_id'].values[0]
        indexproductid = df_sanpham[df_sanpham['Product_id'] == product_id].index[0]

        similarProduct = list(enumerate(similar[indexproductid]))

        sortedSimilarProduct = sorted(similarProduct, key=lambda x: x[1],
reverse=True)

    def lay_ten(index):
        return (df_sanpham[df_sanpham.index == index]['Product_name'].values[0])

    for i in range(1, number + 1):
        ketqua.append(lay_ten(sortedSimilarProduct[i][0]))

    result = "Sản phẩm gợi ý:\n"
    for product in ketqua:
        result += product + "\n"
    return result
else:
    return "Không tìm thấy sản phẩm"

def get_similar_products_by_category(Product_name):
    ketqua = []
    # Tìm kiếm sản phẩm gần giống

```

```

Product_name_list = df_sanpham['Product_name'].tolist()
best_match = process.extractOne(Product_name, Product_name_list)
if best_match:
    Product_name = best_match[0]
    product_id = df_sanpham[df_sanpham['Product_name'] ==
Product_name]['Product_id'].values[0]
    category = df_sanpham[df_sanpham['Product_id'] ==
product_id]['Category'].values[0]

    # Lấy danh sách sản phẩm trong cùng một danh mục
    similar_products = df_sanpham[df_sanpham['Category'] == category]

    # Loại bỏ sản phẩm đã chọn
    similar_products = similar_products[similar_products['Product_name'] !=
Product_name]

    # Lấy 5 sản phẩm đầu tiên
    similar_products = similar_products.head(5)

    # Lấy danh sách tên sản phẩm
    ketqua = similar_products['Product_name'].tolist()

    result = "Sản phẩm gợi ý:\n"
    for product in ketqua:
        result += product + "\n"
    return result
else:
    return "Không tìm thấy sản phẩm"

def collaborative_filtering(customer_id):
    # Đọc dữ liệu từ cơ sở dữ liệu
    query = 'SELECT * FROM Ratings'
    ratings = pd.read_sql(query, conn)

    # Tạo ma trận đánh giá trung bình
    user_item_matrix = ratings.pivot(index='Customer_id', columns='Product_id',
values='Rating_avg')

    # Tạo mô hình nearest neighbors
    model = NearestNeighbors(n_neighbors=5, algorithm='brute', metric='cosine')
    model.fit(user_item_matrix.fillna(0))

    # Tìm kiếm sản phẩm gợi ý cho khách hàng

```

```
distances, indices = model.kneighbors(user_item_matrix.iloc[customer_id-1].values.reshape(1, -1), n_neighbors=5)
```

```
# Lấy danh sách sản phẩm gợi ý ``python
```

```
# Lấy danh sách sản phẩm gợi ý
```

```
recommended_products = []
```

```
for index in indices[0]:
```

```
    product_id = user_item_matrix.columns[index]
```

```
        product_name = df_sanpham[df_sanpham['Product_id'] ==
```

```
product_id]['Product_name'].values[0]
```

```
    recommended_products.append(product_name)
```

```
# Trả về danh sách sản phẩm gợi ý
```

```
result = "Sản phẩm gợi ý:\n"
```

```
for product in recommended_products:
```

```
    result += product + "\n"
```

```
return result
```

```
demo = gr.Interface(
```

```
    fn=get_similar_products_by_category,
```

```
    inputs=gr.Textbox(label="Nhập tên sản phẩm"),
```

```
    outputs=gr.Textbox(label="Kết quả"),
```

```
    title="Gợi ý sản phẩm",
```

```
    description="Nhập tên sản phẩm để nhận gợi ý"
```

```
)
```

```
demo.launch()
```

4. Kết Luận:

Gợi ý sản phẩm

Nhập tên sản phẩm để nhận gợi ý

Nhập tên sản phẩm

Tai nghe

Clear

Submit

Kết quả

Sản phẩm gợi ý:

Tai nghe chụp tai Logitech H150 - 2 jack 3.5mm, Mic khử giảm tiếng ồn, âm thanh nổi - Hàng chính hãng

Chuột không dây Logitech M185 - Hàng chính hãng

Chuột Không Dây Logitech M187 - Hàng chính hãng

Loa Vi Tính SoundMax A-130/2.0 6W - Hàng Chính Hãng

Combo chuột phím không dây Logitech MK240 - Hàng chính hãng

Flag

Hình 71: Giao diện hệ thống đề xuất sản phẩm.

Trong giai đoạn này, nhóm đã xây dựng một hệ thống gợi ý sản phẩm tự động sử dụng machine learning trên bộ dữ liệu đã có. Một ví dụ khi khách hàng sử dụng giao diện tìm kiếm sản phẩm, mô hình sẽ hiển thị kết quả gợi ý như sau:

Sản phẩm nhập vào: Tai nghe

Danh sách sản phẩm gợi ý:

1. Tai nghe chụp tai Logitech H150 - 2 jack 3.5mm, Mic khử giảm tiếng ồn, âm thanh nổi - Hàng chính hãng
2. Chuột không dây Logitech M185 - Hàng chính hãng
3. Chuột Không Dây Logitech M187 - Hàng chính hãng
4. Loa Vi Tính SoundMax A-130/2.0 6W - Hàng Chính Hãng
5. Combo chuột phím không dây Logitech MK240 - Hàng chính hãng

Qua đó cho thấy Hệ thống đã hoạt động đúng yêu cầu, điều này sẽ giúp cải thiện trải nghiệm dịch vụ cũng như, tăng thêm tỷ lệ chuyển đổi cũng như giúp quá trình Cross Sale của doanh nghiệp sẽ trở nên dễ dàng hơn vì mô hình sẽ gợi ý các sản phẩm có liên quan nhất trong cùng 1 mặt hàng hoặc trong cùng 1 doanh mục sản phẩm.